

日 本 国 特 許 庁
JAPAN PATENT OFFICE

別紙添付の書類に記載されている事項は下記の出願書類に記載されている事項と同一であることを証明する。

This is to certify that the annexed is a true copy of the following application as filed with this Office.

出 願 年 月 日
Date of Application: 2 0 0 5 年 5 月 2 5 日

出 願 番 号
Application Number: 特 願 2 0 0 5 - 1 5 2 7 8 8

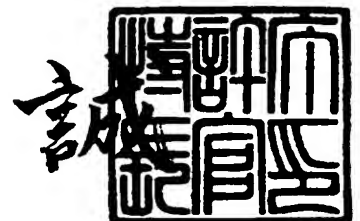
パリ条約による外国への出願
に用いる優先権の主張の基礎
となる出願の国コードと出願
番号
J P 2 0 0 5 - 1 5 2 7 8 8
The country code and number
of your priority application,
to be used for filing abroad
under the Paris Convention, is

出 願 人
Applicant(s): ソニー株式会社

2 0 0 5 年 1 0 月 2 6 日

特許庁長官
Commissioner,
Japan Patent Office.

中 嶋



BEST AVAILABLE COPY

【官 報 号】 付 訂 願
【整理番号】 0500017605
【提出日】 平成17年 5月25日
【あて先】 特許庁長官殿
【国際特許分類】 G06F 15/16
【発明者】
 【住所又は居所】 東京都港区南青山二丁目6番21号 株式会社ソニー・コンピュ
 ータエンタテインメント内
 【氏名】 鈴鹿 倫之
【特許出願人】
 【識別番号】 000002185
 【氏名又は名称】 ソニー株式会社
【代理人】
 【識別番号】 100093241
 【弁理士】
 【氏名又は名称】 宮田 正昭
 【電話番号】 03-5541-7577
【選任した代理人】
 【識別番号】 100101801
 【弁理士】
 【氏名又は名称】 山田 英治
 【電話番号】 03-5541-7577
【選任した代理人】
 【識別番号】 100086531
 【弁理士】
 【氏名又は名称】 澤田 俊夫
 【電話番号】 03-5541-7577
【先の出願に基づく優先権主張】
 【出願番号】 特願2004-283531
 【出願日】 平成16年 9月29日
【手数料の表示】
 【予納台帳番号】 048747
 【納付金額】 16,000円
【提出物件の目録】
 【物件名】 特許請求の範囲 1
 【物件名】 明細書 1
 【物件名】 図面 1
 【物件名】 要約書 1
 【包括委任状番号】 9904833

【請求項 1】

複数のオペレーティングシステム（OS）を格納した情報処理装置であり、
複数OS間の通信制御を実行する制御OSを有し、
前記制御OSは、

通信実行OS各々に対応して設定される論理パーティション相互のメッセージ転送制御
を実行する構成であり、物理アドレス空間のメッセージ領域を、メッセージ送信側OSの
論理パーティションアドレス空間のメッセージ領域へのマッピング状態から、メッセージ
受信側OSの論理パーティションアドレス空間のメッセージ領域へのマッピング状態に切
り替える処理を実行して通信実行OS間のメッセージ転送制御を行なう構成であることを
特徴とする情報処理装置。

【請求項 2】

前記通信実行OS中の少なくとも1つの通信実行OSは、

自己の管理するファイルシステムによって識別可能なファイルディスクリプタに関連付
けられたソケットを生成し、該ソケットを介してアクセス可能な仮想ファイルを作成し、
前記物理アドレス空間のメッセージ領域を該仮想ファイルを介してアクセスする構成であ
ることを特徴とする請求項 1 に記載の情報処理装置。

【請求項 3】

ソケットを適用した通信を実行する通信実行OSは、

ソケットに対応付けられた仮想ファイルの識別子を取得し、取得した仮想ファイル識別
子によって特定される仮想ファイルを適用したメッセージ書き込みまたはメッセージ読み
取りを実行する構成であることを特徴とする請求項 2 に記載の情報処理装置。

【請求項 4】

前記通信実行OS中の少なくとも1つの通信実行OSは、

自己の管理するファイルシステムによって識別可能なファイルディスクリプタに関連付
けられたソケットを生成し、該ソケットを介して前記物理アドレス空間のメッセージ領域
をプロセスのアドレス空間へマッピングし、プロセスが直接メッセージ領域へのアクセス
を実行する構成であることを特徴とする請求項 1 に記載の情報処理装置。

【請求項 5】

前記通信実行OS中の少なくとも1つの通信実行OSは、

自己の管理するファイルシステムによって識別可能なファイルディスクリプタに関連付
けられたソケットを生成し、該ソケットを介して前記物理アドレス空間のメッセージ領域
を前記通信実行OSに対応する論理パーティションのアドレス空間へマッピングし、前記
メッセージ領域へのアクセスを実行する構成であることを特徴とする請求項 1 に記載の情
報処理装置。

【請求項 6】

ソケットを適用した通信を実行する通信実行OSは、

前記ソケットに対応するサービス識別子を設定し、サービス対応の通信許可設定を実行
する構成であることを特徴とする請求項 2 乃至 5 いずれかに記載の情報処理装置。

【請求項 7】

ソケットを適用した通信を実行する通信実行OSは、

前記ソケットを介したメッセージの受信監視処理を実行する構成であることを特徴とす
る請求項 2 乃至 5 いずれかに記載の情報処理装置。

【請求項 8】

前記通信実行OSは、

システムコールとしてのセレクトシステムコールの適用により、前記ソケットを介した
メッセージの受信監視処理を実行する構成であることを特徴とする請求項 7 に記載の情報
処理装置。

【請求項 9】

複数のオペレーティングシステム（OS）を格納した情報処理装置における通信処理方

広げること、

物理アドレス空間のメッセージ領域を、メッセージ送信側OSの論理パーティションアドレス空間のメッセージ領域へのマッピング状態とするステップと、

前記マッピング状態を解除し、物理アドレス空間のメッセージ領域を、メッセージ受信側OSの論理パーティションアドレス空間のメッセージ領域へのマッピング状態に変更するステップと、

を有することを特徴とする通信処理方法。

【請求項10】

メッセージ転送を実行する少なくとも1つの通信実行OSは、

自己の管理するファイルシステムによって識別可能なファイルディスクリプタに関連付けられたソケットを生成し、該ソケットを介してアクセス可能な仮想ファイルを作成し、前記物理アドレス空間のメッセージ領域を該仮想ファイルを介してアクセスすることによるメッセージ送受信を行なうことを特徴とする請求項9に記載の通信処理方法。

【請求項11】

ソケットを適用した通信を実行する通信実行OSは、

ソケットに対応付けられた仮想ファイルの識別子を取得し、取得した仮想ファイル識別子によって特定される仮想ファイルを適用したメッセージ書き込みまたはメッセージ読み取りを実行することを特徴とする請求項10に記載の通信処理方法。

【請求項12】

前記通信実行OS中の少なくとも1つの通信実行OSは、

自己の管理するファイルシステムによって識別可能なファイルディスクリプタに関連付けられたソケットを生成し、該ソケットを介して前記物理アドレス空間のメッセージ領域をプロセスのアドレス空間へマッピングし、プロセスが直接メッセージ領域へのアクセスを実行することを特徴とする請求項9に記載の通信処理方法。

【請求項13】

前記通信実行OS中の少なくとも1つの通信実行OSは、

自己の管理するファイルシステムによって識別可能なファイルディスクリプタに関連付けられたソケットを生成し、該ソケットを介して前記物理アドレス空間のメッセージ領域を前記通信実行OSに対応する論理パーティションのアドレス空間へマッピングし、前記メッセージ領域へのアクセスを実行する構成であることを特徴とする請求項9に記載の通信処理方法。

【請求項14】

ソケットを適用した通信を実行する通信実行OSは、

前記ソケットに対応するサービス識別子を設定し、サービス対応の通信許可設定を実行することを特徴とする請求項9乃至13いずれかに記載の通信処理方法。

【請求項15】

ソケットを適用した通信を実行する通信実行OSは、

前記ソケットを介したメッセージの受信監視処理を実行することを特徴とする請求項9乃至13いずれかに記載の通信処理方法。

【請求項16】

前記通信実行OSは、

システムコールとしてのセレクトシステムコールの適用により、前記ソケットを介したメッセージの受信監視処理を実行することを特徴とする請求項15に記載の通信処理方法。

【請求項17】

複数のオペレーティングシステム(OS)を格納した情報処理装置における通信処理制御を実行するコンピュータ・プログラムであり、

物理アドレス空間のメッセージ領域を、メッセージ送信側OSの論理パーティションアドレス空間のメッセージ領域へのマッピング状態とするステップと、

前記マッピング状態を解除し、物理アドレス空間のメッセージ領域を、メッセージ受信

開のこの調理バーフィッシュンアドレス空間のメッセージ領域へのマッピング状態に変わるステップと、

を有することを特徴とするコンピュータ・プログラム。

【請求項 18】

複数のオペレーティングシステム（OS）を格納した情報処理装置における通信処理制御を実行するコンピュータ・プログラムであり、

通信実行OSの管理するファイルシステムによって識別可能なファイルディスクリプタに関連付けられたソケットを生成するステップと、

前記ソケットを介してアクセス可能な仮想ファイルを作成するステップと、

前記物理アドレス空間のメッセージ領域を前記仮想ファイルを介してアクセスすることによってメッセージ送受信を行なうステップと、

を有することを特徴とするコンピュータ・プログラム。

【請求項 19】

複数のオペレーティングシステム（OS）を格納した情報処理装置における通信処理制御を実行するコンピュータ・プログラムであり、

通信実行OSの管理するファイルシステムによって識別可能なファイルディスクリプタに関連付けられたソケットを生成するステップと、

前記ソケットを介して前記物理アドレス空間のメッセージ領域をプロセスのアドレス空間へマッピングするステップと、

プロセスが直接メッセージ領域へのアクセスを実行するステップと、

を有することを特徴とするコンピュータ・プログラム。

【発明の名称】 情報処理装置、通信処理方法、並びにコンピュータ・プログラム

【技術分野】

【0001】

本発明は、情報処理装置、通信処理方法、並びにコンピュータ・プログラムに関する。さらに詳細には、複数のオペレーティングシステム（OS）によるプロセス実行を可能としたマルチOS構成において、OS間のメッセージ転送制御を実行する情報処理装置、通信処理方法、並びにコンピュータ・プログラムに関する。

【背景技術】

【0002】

1つのシステム内に複数のオペレーティングシステム（OS）を搭載したマルチOSシステムにおいては、各OSはそれぞれ異なるプロセスが実行可能であり、システムで共通のハードウェア、すなわちCPUやメモリ等を時系列に順次切り替えて利用した処理が行なわれる。

【0003】

複数OSの各々の実行プロセス（タスク）のスケジューリングは、例えばパーティション管理ソフトウェアによって実行される。1つのシステムにOS（ α ）とOS（ β ）の2つのオペレーティングシステムが並存する場合、OS（ α ）の処理をパーティションAとし、OS（ β ）の処理をパーティションBとすると、パーティション管理ソフトウェアは、パーティションAとパーティションBの実行スケジュールを決定し、決定したスケジュールに基づいて、ハードウェア資源を割り当てて各OSにおける処理を実行する。

【0004】

マルチOS型のシステムにおけるタスク管理を開示した従来技術として、例えば、特許文献1がある。特許文献1には、複数のOSの各々において実行されるタスク管理において、緊急性の高い処理を優先的に処理させるためのタスクスケジューリング手法を開示している。

【0005】

複数OSが共存するシステムでは、OS間のメッセージ転送を行なう場合、OS間を接続するメッセージチャネルを生成し、通信を実行するOS各々がメッセージチャネル接続の通信用インタフェースを設定することが必要となる。メッセージチャネルを介した通信を実行する通信実行OSが様々な異なるOSである場合、通信インタフェース仕様の制限や、異なるOS間通信に起因するオーバーヘッドの発生など、様々な問題がある。

【特許文献1】 特開2003-345612号公報

【発明の開示】

【発明が解決しようとする課題】

【0006】

本発明は、上述の問題点に鑑みてなされたものであり、複数OSが共存するマルチOS環境におけるOS間通信を余分なオーバーヘッドを発生させることなくスムーズに実行することを可能とした情報処理装置、通信処理方法、並びにコンピュータ・プログラムを提供することを目的とする。

【課題を解決するための手段】

【0007】

本発明の第1の側面は、

複数のオペレーティングシステム（OS）を格納した情報処理装置であり、

複数OS間の通信制御を実行する制御OSを有し、

前記制御OSは、

通信実行OS各々に対応して設定される論理パーティション相互のメッセージ転送制御を実行する構成であり、物理アドレス空間のメッセージ領域を、メッセージ送信側OSの論理パーティションアドレス空間のメッセージ領域へのマッピング状態から、メッセージ受信側OSの論理パーティションアドレス空間のメッセージ領域へのマッピング状態に切

ソケットを介して通信を行う間のプロセス転送制御を行う構成であることを特徴とする情報処理装置にある。

【0008】

さらに、本発明の情報処理装置の一実施態様において、前記通信実行OS中の少なくとも1つの通信実行OSは、自己の管理するファイルシステムによって識別可能なファイルディスクリプタに関連付けられたソケットを生成し、該ソケットを介してアクセス可能な仮想ファイルを作成し、前記物理アドレス空間のメッセージ領域を該仮想ファイルを介してアクセスする構成であることを特徴とする。

【0009】

さらに、本発明の情報処理装置の一実施態様において、ソケットを適用した通信を実行する通信実行OSは、ソケットに対応付けられた仮想ファイルの識別子を取得し、取得した仮想ファイル識別子によって特定される仮想ファイルを適用したメッセージ書き込みまたはメッセージ読み取りを実行する構成であることを特徴とする。

【0010】

さらに、本発明の情報処理装置の一実施態様において、前記通信実行OS中の少なくとも1つの通信実行OSは、自己の管理するファイルシステムによって識別可能なファイルディスクリプタに関連付けられたソケットを生成し、該ソケットを介して前記物理アドレス空間のメッセージ領域をプロセスのアドレス空間へマッピングし、プロセスが直接メッセージ領域へのアクセスを実行する構成であることを特徴とする。

【0011】

さらに、本発明の情報処理装置の一実施態様において、前記通信実行OS中の少なくとも1つの通信実行OSは、自己の管理するファイルシステムによって識別可能なファイルディスクリプタに関連付けられたソケットを生成し、該ソケットを介して前記物理アドレス空間のメッセージ領域を前記通信実行OSに対応する論理パーティションのアドレス空間へマッピングし、前記メッセージ領域へのアクセスを実行する構成であることを特徴とする。

【0012】

さらに、本発明の情報処理装置の一実施態様において、ソケットを適用した通信を実行する通信実行OSは、前記ソケットに対応するサービス識別子を設定し、サービス対応の通信許可設定を実行する構成であることを特徴とする。

【0013】

さらに、本発明の情報処理装置の一実施態様において、ソケットを適用した通信を実行する通信実行OSは、前記ソケットを介したメッセージの受信監視処理を実行する構成であることを特徴とする。

【0014】

さらに、本発明の情報処理装置の一実施態様において、前記通信実行OSは、システムコールとしてのセレクトシステムコールの適用により、前記ソケットを介したメッセージの受信監視処理を実行する構成であることを特徴とする。

【0015】

さらに、本発明の第2の側面は、

複数のオペレーティングシステム(OS)を格納した情報処理装置における通信処理方法であり、

物理アドレス空間のメッセージ領域を、メッセージ送信側OSの論理パーティションアドレス空間のメッセージ領域へのマッピング状態とするステップと、

前記マッピング状態を解除し、物理アドレス空間のメッセージ領域を、メッセージ受信側OSの論理パーティションアドレス空間のメッセージ領域へのマッピング状態に変更するステップと、

を有することを特徴とする通信処理方法にある。

【0016】

さらに、本発明の通信処理方法の一実施態様において、メッセージ転送を実行する少な

、ととも１つの通信実行方法は、自己の管理するファイルシステムによって識別可能なファイルディスクリプタに関連付けられたソケットを生成し、該ソケットを介してアクセス可能な仮想ファイルを作成し、前記物理アドレス空間のメッセージ領域を該仮想ファイルを介してアクセスすることによるメッセージ送受信を行なうことを特徴とする。

【００１７】

さらに、本発明の通信処理方法の一実施態様において、ソケットを適用した通信を実行する通信実行ＯＳは、ソケットに対応付けられた仮想ファイルの識別子を取得し、取得した仮想ファイル識別子によって特定される仮想ファイルを適用したメッセージ書き込みまたはメッセージ読み取りを実行することを特徴とする。

【００１８】

さらに、本発明の通信処理方法の一実施態様において、前記通信実行ＯＳ中の少なくとも１つの通信実行ＯＳは、自己の管理するファイルシステムによって識別可能なファイルディスクリプタに関連付けられたソケットを生成し、該ソケットを介して前記物理アドレス空間のメッセージ領域をプロセスのアドレス空間へマッピングし、プロセスが直接メッセージ領域へのアクセスを実行することを特徴とする。

【００１９】

さらに、本発明の通信処理方法の一実施態様において、前記通信実行ＯＳ中の少なくとも１つの通信実行ＯＳは、自己の管理するファイルシステムによって識別可能なファイルディスクリプタに関連付けられたソケットを生成し、該ソケットを介して前記物理アドレス空間のメッセージ領域を前記通信実行ＯＳに対応する論理パーティションのアドレス空間へマッピングし、前記メッセージ領域へのアクセスを実行する構成であることを特徴とする。

【００２０】

さらに、本発明の通信処理方法の一実施態様において、ソケットを適用した通信を実行する通信実行ＯＳは、前記ソケットに対応するサービス識別子を設定し、サービス対応の通信許可設定を実行することを特徴とする。

【００２１】

さらに、本発明の通信処理方法の一実施態様において、ソケットを適用した通信を実行する通信実行ＯＳは、前記ソケットを介したメッセージの受信監視処理を実行することを特徴とする。

【００２２】

さらに、本発明の通信処理方法の一実施態様において、前記通信実行ＯＳは、システムコールとしてのセレクトシステムコールの適用により、前記ソケットを介したメッセージの受信監視処理を実行することを特徴とする。

【００２３】

さらに、本発明の第３の側面は、

複数のオペレーティングシステム（ＯＳ）を格納した情報処理装置における通信処理制御を実行するコンピュータ・プログラムであり、

物理アドレス空間のメッセージ領域を、メッセージ送信側ＯＳの論理パーティションアドレス空間のメッセージ領域へのマッピング状態とするステップと、

前記マッピング状態を解除し、物理アドレス空間のメッセージ領域を、メッセージ受信側ＯＳの論理パーティションアドレス空間のメッセージ領域へのマッピング状態に変更するステップと、

を有することを特徴とするコンピュータ・プログラムにある。

【００２４】

さらに、本発明の第４の側面は、

複数のオペレーティングシステム（ＯＳ）を格納した情報処理装置における通信処理制御を実行するコンピュータ・プログラムであり、

通信実行ＯＳの管理するファイルシステムによって識別可能なファイルディスクリプタに関連付けられたソケットを生成するステップと、

前記ソケットを介してプロセスへ可能な仮想ファイルを作成するステップと、
前記物理アドレス空間のメッセージ領域を前記仮想ファイルを介してアクセスすることによってメッセージ送受信を行なうステップと、
を有することを特徴とするコンピュータ・プログラムにある。

【0025】

さらに、本発明の第5の側面は、
複数のオペレーティングシステム（OS）を格納した情報処理装置における通信処理制御を実行するコンピュータ・プログラムであり、
通信実行OSの管理するファイルシステムによって識別可能なファイルディスクリプタに関連付けられたソケットを生成するステップと、
前記ソケットを介して前記物理アドレス空間のメッセージ領域をプロセスのアドレス空間へマッピングするステップと、
プロセスが直接メッセージ領域へのアクセスを実行するステップと、
を有することを特徴とするコンピュータ・プログラムにある。

【0026】

なお、本発明のコンピュータ・プログラムは、例えば、様々なプログラム・コードを実行可能な汎用コンピュータ・システムに対して、コンピュータ可読な形式で提供する記憶媒体、通信媒体、例えば、CDやFD、MOなどの記憶媒体、あるいは、ネットワークなどの通信媒体によって提供可能なコンピュータ・プログラムである。このようなプログラムをコンピュータ可読な形式で提供することにより、コンピュータ・システム上でプログラムに応じた処理が実現される。

【0027】

本発明のさらに他の目的、特徴や利点は、後述する本発明の実施例や添付する図面に基づく、より詳細な説明によって明らかになるであろう。なお、本明細書においてシステムとは、複数の装置の論理的集合構成であり、各構成の装置が同一筐体内にあるものには限らない。

【発明の効果】

【0028】

本発明の構成によれば、複数のオペレーティングシステム（OS）が共存するマルチOS環境におけるOS間通信において、OS間の通信制御を実行する制御OSを設定し、制御OSが、通信実行OS各々に対応して設定される論理パーティション相互のメッセージ転送制御を実行する構成とし、物理アドレス空間のメッセージ領域を、メッセージ送信側OSの論理パーティションアドレス空間のメッセージ領域へのマッピング状態から、メッセージ受信側OSの論理パーティションアドレス空間のメッセージ領域へのマッピング状態に切り替える処理を実行してOS間のメッセージ転送制御を行なう構成としたので、異なるOS間の通信におけるスムーズな通信が可能となる。

【0029】

さらに、本発明の構成によれば、通信実行OSが、自己の管理するファイルシステムによって識別可能なファイルディスクリプタに関連付けられたソケットを生成し、ソケットを介してアクセス可能な仮想ファイルを作成し、物理アドレス空間のメッセージ領域を該仮想ファイルを介してアクセスすることで、メッセージ送受信を実行する構成としたので、異なるOSとの通信において汎用的なソケットを適用したメッセージパッシングが実現される。

【0030】

さらに、本発明の構成によれば、通信実行OSが、自己の管理するファイルシステムによって識別可能なファイルディスクリプタに関連付けられたソケットを生成し、該ソケットを介して前記物理アドレス空間のメッセージ領域をプロセスのアドレス空間へマッピングし、プロセスが直接メッセージ領域へのアクセスを実行する構成としたので、異なるOSとの通信において汎用的なソケットを適用したメッセージパッシングが実現されると同時に、少ないオーバーヘッドでプロセスがメッセージ領域にアクセスすることが実現される。

【0031】

さらに、本発明の構成によれば、ソケットに対応するサービス識別子を設定することにより、サービス対応の通信許可設定が可能となり、また、例えばシステムコールとしてのセレクトシステムコールの適用により、ソケットを介したメッセージの受信監視処理が可能となる。

【発明を実施するための最良の形態】

【0032】

以下、図面を参照しながら、本発明の情報処理装置、通信処理方法、並びにコンピュータ・プログラムの詳細について説明する。

【0033】

まず、図1を参照して、本発明の情報処理装置のハードウェア構成例について説明する。プロセッサモジュール101は、複数のプロセッサ(Processing Unit)から構成されたモジュールであり、ROM(Read Only Memory)104、HDD123等に記憶されているプログラムに従って、オペレーティングシステム(OS: Operating System)、OS対応のアプリケーション・プログラムなど、各種プログラムに従ったデータ処理を実行する。プロセッサモジュール101の詳細については、さらに、後段で、図2を参照して説明する。

【0034】

グラフィックエンジン102は、プロセッサモジュール101から入力される指示に従って、出力部122を構成する表示デバイスに画面出力するためのデータ生成、例えば3Dグラフィック描画処理を実行する。メインメモリ(DRAM)103には、プロセッサモジュール101において実行するプログラムやその実行において適宜変化するパラメータ等を格納する。これらはCPUバスなどから構成されるホストバス111により相互に接続されている。

【0035】

ホストバス111は、ブリッジ105を介して、PCI(Peripheral Component Interconnect/Interface)バスなどの外部バス112に接続されている。ブリッジ105は、ホストバス111、外部バス112間、およびコントローラ106、メモリカード107、その他のデバイスとのデータ入出力制御を実行する。

【0036】

入力部121は、キーボード、ポインティングデバイスなどのユーザにより操作される入力デバイスからの入力情報を入力する。出力部122は、液晶表示装置やCRT(Cathode Ray Tube)などの画像出力部とスピーカ等からなる音声出力部から構成される。

【0037】

HDD(Hard Disk Drive)123は、ハードディスクを内蔵し、ハードディスクを駆動し、プロセッサモジュール101によって実行するプログラムや情報を記録または再生させる。

【0038】

ドライブ124は、装着されている磁気ディスク、光ディスク、光磁気ディスク、または半導体メモリ等のリムーバブル記録媒体127に記録されているデータまたはプログラムを読み出して、そのデータまたはプログラムを、インタフェース113、外部バス112、ブリッジ105、およびホストバス111を介して接続されているメインメモリ(DRAM)103に供給する。

【0039】

接続ポート125は、外部接続機器128を接続するポートであり、USB、IEEE1394等の接続部を持つ。接続ポート125は、インタフェース113、外部バス112、ブリッジ105、およびホストバス111を介してプロセッサモジュール101等に接続されている。通信部126は、ネットワークに接続され、プロセッサモジュール101や、HDD123等から提供されるデータの送信、外部からのデータ受信を実行する。

【0040】

次に、プロセッサモジュールの構成例について、図2を参照して説明する。図2に示すように、プロセッサモジュール200は、複数のメインプロセッサからなるメインプロセッサグループ201、複数のサブプロセッサからなる複数のサブプロセッサグループ202～20nによって構成されている。それぞれにメモリコントローラ、2次キャッシュが設けられている。各プロセッサグループ201～20nの各々は例えば8つのプロセッサユニットを有し、クロスバーアーキテクチャ、あるいはバケット交換式ネットワークなどによって接続されている。メインプロセッサグループ201のメインプロセッサの指示のもとに、複数のサブプロセッサグループ202～20nの1以上のサブプロセッサが選択され、所定のプログラムが実行される。

【0041】

各プロセッサグループに設置されたメモリフローコントローラは、図1に示すメインメモリ103とのデータ入出力制御を実行し、2次キャッシュは、各プロセッサグループにおける処理用データの記憶領域として利用される。

【0042】

次に、図3を参照して、本発明の情報処理装置におけるオペレーティングシステム(OS)構成を説明する。本発明の情報処理装置は複数のオペレーティングシステム(OS)が共存するマルチOS構成を持つ。図3に示すように、論理的な階層構成を持つ複数のオペレーティングシステム(OS)を持つ。

【0043】

図3に示すように、下位レイヤに制御OS301を有し、上位レイヤに複数のゲストOS302, 303、およびシステム制御OS304が設定される。制御OS301は、システム制御OS304とともに図1、図2を参照して説明したプロセッサモジュールにおいて実行する1以上のプロセスによって適用される論理パーティションを実現し、システム内のハードウェア資源(計算機資源としてのメインプロセッサ、サブプロセッサ、メモリ、デバイス等)を各論理パーティションに割り振る処理を実行する。

【0044】

ゲストOS302, 303は、例えばゲームOSやWindows(登録商標)、Linux(登録商標)などの各種のOSであり、制御OS301の制御の下に動作するOSである。なお、図3には、ゲストOS302, 303の2つのゲストOSのみを示してあるが、ゲストOSは任意の数に設定することが可能である。

【0045】

ゲストOS302, 303は、制御OS301およびシステム制御OS304によって設定された論理パーティション内で動作し、その論理パーティションに割り当てられたメインプロセッサ、サブプロセッサ、メモリ、デバイス等のハードウェア資源を適用して各種のデータ処理を実行する。

【0046】

例えば、ゲストOS(a)302は、制御OS301およびシステム制御OS304によって設定された論理パーティション2に割り当てられたメインプロセッサ、サブプロセッサ、メモリ、デバイス等のハードウェア資源を適用して、ゲストOS(a)302対応のアプリケーション・プログラム305を実行する。また、ゲストOS(b)303は、論理パーティションnに割り当てられたメインプロセッサ、サブプロセッサ、メモリ、デバイス等のハードウェア資源を適用して、ゲストOS(b)303対応のアプリケーション・プログラム306を実行する。制御OS301は、ゲストOSの実行に必要なインタフェースとしてゲストOSプログラミングインタフェースを提供する。

【0047】

システム制御OS304は、論理パーティション管理情報を含むシステム制御プログラム307を生成し、システム制御プログラム307に基づくシステムの動作制御を制御OS301とともに実行する。システム制御プログラム307は、システム制御プログラム・プログラミングインタフェースを用いてシステムのポリシを制御するプログラムである。

。システム制御プログラム・プログラミングインターフェースは、制御OS 301がシステム制御OS 304に提供される。例えばリソース配分の上限值を設定するなど、プログラムによる柔軟なカスタマイズのための手段を提供するのがシステム制御プログラム307の役割である。

【0048】

システム制御プログラム307はシステム制御プログラム・プログラミングインターフェースを用いてシステムの振る舞いを制御することができる。例えば、新しく論理パーティションを作成し、その論理パーティションで新しいゲストOSを起動することができる。複数のゲストOSが動作するシステムでは、それらのゲストOSはシステム制御プログラムにあらかじめプログラムされた順序で起動されることになる。また、ゲストOSから提出された資源割り当て要求を制御OS 301が受理する前に検査し、システムのポリシーに従って修正したり、要求そのものを拒否したりすることができる。これにより、特定のゲストOSだけが資源を独占することがないようにすることができる。このように、システムのポリシーをプログラムとして実現したものがシステム制御プログラムである。

【0049】

制御OS 301はシステム制御OS 304のために特別な論理パーティション（図では論理パーティション1）を割り当てる。制御OS 301は、ハイババイザモードで動作する。ゲストOSはスーパーバイザモードで動作する。システム制御OS、アプリケーション・プログラムはプロブレムモード（ユーザモード）で動作する。

【0050】

論理パーティションはシステム内の資源分配を受ける主体である。たとえばメインメモリ103（図1参照）はいくつかの領域へ分割され、それぞれの領域の使用権が論理パーティションに対して与えられる。論理パーティションに分配される資源の種別には以下のものがある。

- a) 物理プロセッサユニット使用時間
- b) 仮想アドレス空間
- c) 論理パーティション内で動作するプログラムがアクセスできるメモリ
- d) 制御OSが論理パーティションの管理のために用いるメモリ
- e) イベントポート
- f) デバイスの使用権
- g) キャッシュパーティション
- h) バス使用権

【0051】

前述したように、ゲストOSは論理パーティションの中で動作する。ゲストOSは論理パーティションに割り当てられた資源を独占して各種のデータ処理を実行する。多くの場合、システム上で動作する個々のゲストOS毎に1つの論理パーティションが作成される。各論理パーティションにはユニークな識別子が与えられる。システム制御OS 304は、論理パーティション管理情報として生成するシステム制御プログラムに識別子を対応づけて管理する。

【0052】

論理パーティションは、制御OS 301およびシステム制御OS 304によって生成される。生成直後の論理パーティションは何も資源を持たず、使用資源の制限も設定されていない。論理パーティションには活動状態と終了状態という2つの状態がある。生成直後の論理パーティションは活動状態にある。論理パーティション内で動作するゲストOSの要求に基づいて論理パーティションは終了状態に遷移し、論理パーティションに割り当てられている全ての論理プロセッサが停止する。

【0053】

なお、論理プロセッサは、論理パーティションに割り当てられる論理的なプロセッサであり、いずれかの物理プロセッサ、すなわち、図2に示すプロセッサグループ内のプロセッサに対応する。ただし、論理プロセッサと物理プロセッサは必ずしも1対1の関係には

なく、1つの論理プロセスに複数の物理プロセスが対応付けられる場合もあり、複数の論理プロセスに1つの物理プロセスが対応付けられる場合もある。論理プロセスと物理プロセスの対応付けは、制御OS301が決定する。

【0054】

制御OS301は、各論理パーティションが使用する資源の量を制限する機能を備えている。ゲストOS302, 303がシステム制御OS304との通信を行うことなく割り当て／解放が行える資源については使用量の制限が可能となっている。

【0055】

各論理パーティションは制御シグナルポートを備えている。このポートには論理パーティション間のデータ交換／共有に必要な様々な制御シグナルが到着する。制御シグナルの例を以下に挙げる。

- a) 論理パーティション間イベントポートの接続依頼
- b) 論理パーティション間メッセージチャネルの接続依頼
- c) 共有メモリ領域への接続依頼

【0056】

各論理パーティションに到着した制御シグナルは制御シグナルポートでキューイングされる。キューの深さは、メモリ資源が許す範囲であれば、制限は無い。キューイングに必要なメモリ資源は制御シグナルを送った論理パーティションから確保される。このポートから制御シグナルを取り出すためには、ゲストOSプログラミングインタフェースを呼び出す。空の制御シグナルポートに制御シグナルが到着したときに、任意のイベントポートにイベントを送信することが可能である。イベントポートの指定はゲストOSプログラミングインタフェースを呼び出すことによって行える。

【0057】

次に、図4以下の各図を参照して、OS間通信について説明する。前述したように、各ゲストOS、およびシステム制御OSの各々に対しては、論理パーティションが対応付けられる。各論理パーティションモジュール間のメッセージ転送は、制御OSの生成するメッセージチャネルを介して実行される。以下、

- (1) ゲストOS間の通信処理
 - (2) ゲストOSとシステム制御OS間の通信処理
- の2つの通信処理について順次説明する。

【0058】

〔(1) ゲストOS間の通信処理〕

まず、ゲストOS間の通信処理について説明する。図4は、制御OS410上の2つのゲストOS間の通信処理を説明する概念図である。ゲストOS(A)420からゲストOS(B)430にメッセージを転送する処理例を示している。

【0059】

制御OS410は、ゲストOS間の通信、あるいはゲストOSとシステム制御OS間の通信用のメッセージチャネル412を設定する。メッセージチャネルは2つの論理パーティション間のデータ転送機構として設定されるものであり、コネクションオリエンテッドな双方向の通信路として設定される。

【0060】

図4を参照して、制御OS410の設定するメッセージチャネル412を介したゲストOS(A)420とゲストOS(B)430間の通信処理について説明する。ゲストOS(A)420をメッセージチャネルのクライアント(接続元)、ゲストOS(B)430をメッセージチャネルのサーバ(接続先)とする。

ゲストOS(A)420はメッセージチャネルの接続要求をゲストOS(B)430に対して行う。

ゲストOS(B)がメッセージチャネルの接続を許可すると、図4に示すように、制御OS410の制御の下にメッセージチャネル412がゲストOS(A)420とゲストOS(B)430の間に接続される。メッセージチャネル412は、通信実行OSとしての

セノヘドに設定されるコネクション端点としてのメッセージポート422, 432を接続するチャンネルとして設定される。

【0061】

なお、ゲストOS(B)430がメッセージチャンネルの接続を拒否すると、メッセージチャンネルの接続は行われない。各ゲストOSに対応するメッセージポート422, 432にはイベント受信ポート421, 431が関連づけて設定される。

【0062】

制御OS410は、OS(論理パーティション)間通信、およびOS(論理パーティション)内通信のためにイベント機構を用意している。イベントはイベント送信ポートからイベント受信ポートへの1対1、かつ単方向のコネクションを通して送られる。イベント受信ポートはイベントをキューイング及びカウントしない。例えばイベント受信ポートに読み出される前のイベントがある状態で新たなイベントが到着した場合、イベント受信ポートの状態はペンディング状態から変化しない。コネクションが確立されたイベント送信ポートは何度でもイベントを送信することができる。イベント受信ポートは論理パーティション内のゲストOSがイベントを受け取るためのポートである。イベント受信ポートに到着するイベントは割り込みの一つとして扱われる。

【0063】

OS(論理パーティション)間にメッセージチャンネルが接続されたり、空のメッセージポートにメッセージが到着すると、関連づけられたイベント受信ポートにイベントが配送される。

【0064】

ゲストOS(B)がメッセージチャンネルの接続を許可すると、図4に示すように、制御OS410の制御の下にメッセージチャンネル412がゲストOS(A)420とゲストOS(B)430の間に接続され、その後、ゲストOS(A)420は送信するメッセージ領域を用意する。なお、メッセージ領域実体としての物理アドレス空間メッセージ領域411はシステムのメモリ上にある。

【0065】

図4に示すゲストOS(A)420のメッセージ領域423と、ゲストOS(B)430のメッセージ領域433は、それぞれのOSに対応する論理パーティションのアドレス空間における論理アドレス空間メッセージ領域である。ある瞬間に、物理アドレス空間メッセージ領域411はただ一つの論理パーティションに属している。物理アドレス空間メッセージ領域411が属していない論理パーティションからのメッセージ領域へのアクセスはできない。

【0066】

すなわち、物理アドレス空間メッセージ領域411は、ある瞬間において、ゲストOS(A)420の論理アドレス空間メッセージ領域423にマッピングされた状態にあるか、あるいは、ゲストOS(B)430の論理アドレス空間メッセージ領域433にマッピングされた状態にあるかいずれか一方の状態にある。このマッピング制御は制御OS410によって実行される。このように、制御OS410は、メッセージチャンネル412を介したメッセージ転送を論理パーティションアドレス空間から物理アドレス空間への仮想的なアドレス変換機構を用いて実現する。

【0067】

図5を参照して、仮想的なアドレス変換機構を用いたメッセージ転送処理について説明する。

【0068】

図5において、(1)状態1、(2)状態2は、ゲストOS(A)からゲストOS(B)に対するメッセージ転送の前後の状態を示している。(1)状態1において、物理アドレス空間450に確保されたメッセージ領域451は、メッセージ転送元(クライアント)側の論理パーティションの論理パーティションアドレス空間、すなわち、ゲストOS(A)の論理パーティションアドレス空間460のメッセージ領域461にマップされてい

る。この状態において、メッセージ領域451が属していない論理パーティション、すなわち、ゲストOS(B)の論理パーティションアドレス空間470のメッセージ領域471から物理アドレス空間450のメッセージ領域451へのアクセスはできない。

【0069】

メッセージ領域451が転送されると、物理アドレス空間450に確保されたメッセージ領域451は、クライアント側論理パーティション、すなわち、ゲストOS(A)の論理パーティションアドレス空間460からアンマップ(マップ解除)され、サーバ側論理パーティションの論理パーティションアドレス空間、すなわち、ゲストOS(B)の論理パーティションアドレス空間470のメッセージ領域471にマップされる。すなわち、図5の(2)状態2に設定される。この状態において、メッセージ領域451が属していない論理パーティション、すなわち、ゲストOS(A)の論理パーティションアドレス空間460から物理アドレス空間450のメッセージ領域451へのアクセスはできない。

【0070】

このように、制御OSは、物理アドレス空間450に確保されたメッセージ領域451を、メッセージチャネルのメッセージポートの設定されたゲストOS(論理パーティション)の論理パーティションアドレス空間の一方にマッピングし、他方からアンマップ(マップ解除)する処理によって、クライアントからのメッセージをサーバがアクセス可能な設定とすることでメッセージ転送を実現している。

【0071】

図6を参照して、ゲストOS間のメッセージチャネルを介したメッセージ転送シーケンスについて説明する。

【0072】

メッセージ送信側OS(クライアント)は、ステップS101において、メッセージ受信側OS(サーバ)に対して接続要求を送出する。接続要求は、ステップS102において、制御OSを介してメッセージ受信側OS(サーバ)に送信される。

【0073】

メッセージ受信側OS(サーバ)は、接続要求の受信に基づいて、ステップS103において、接続を許可すると、メッセージ送信側OS(クライアント)とメッセージ送信側OS(クライアント)それぞれにメッセージポートの設定されたメッセージチャネルを生成する。メッセージチャネルの生成が完了すると、ステップS104において、メッセージポートに関連付けられたイベント受信ポートを介してメッセージチャネルの設定完了を各OSに通知する。

【0074】

メッセージ送信側OS(クライアント)は、ステップS105において、メッセージ領域を確保する。図4の論理アドレス空間メッセージ領域423である。ステップS106において、制御OSは、物理アドレス空間のメッセージ領域をメッセージ送信側OS(クライアント)論理パーティションの論理パーティションアドレス空間にマップする。この処理は、図4において、ゲストOS(A)420の論理アドレス空間メッセージ領域423を物理アドレス空間メッセージ領域411にマップする処理に相当する。

【0075】

ステップS107において、制御OSは、物理アドレス空間のメッセージ領域をメッセージ送信側OS(クライアント)側論理パーティションの論理パーティションアドレス空間からアンマップ(マップ解除)するとともに、メッセージ受信側OS(サーバ)側論理パーティションの論理パーティションアドレス空間にマップする。すなわち、図4において、ゲストOS(A)420の論理アドレス空間メッセージ領域423の物理アドレス空間メッセージ領域411に対するマップを解除し、物理アドレス空間メッセージ領域411をゲストOS(B)430の論理アドレス空間メッセージ領域433にマップする処理を実行する。

【0076】

ステップS108において、メッセージ受信側OS(サーバ)は、メッセージ受信側OS

（サーバ）の論理パーティションの論理パーティションアドレス空間を介して物理アドレス空間のメッセージ領域にアクセスし、メッセージを取得する。

【0077】

ステップS109において、メッセージ受信側OS（サーバ）は、メッセージ領域を破棄する。さらに、ステップS110において、メッセージ受信側OS（サーバ）は、メッセージポートを破棄する。メッセージポートの破棄通知は、ステップS111において、イベント受信ポートを適用してメッセージ送信側OS（クライアント）側に送信され、メッセージ送信側OS（クライアント）側においてステップS112でメッセージポートを破棄することでメッセージチャネルが解消され、メッセージチャネルを介したOS間通信が終了する。

【0078】

〔（2）ゲストOSとシステム制御OS間の通信処理〕

次に、図7および図10を参照して、システム制御OSとゲストOS間の通信処理について説明する。前述したように、システム制御OSは、システム制御プログラムを生成し、例えばシステムに予め定められたリソース配分の上限情報などに基づいてシステム全体の動作制御を行う。

【0079】

ゲストOSは、例えばゲストOSに設定された論理パーティションに対応するリソース要求など様々な処理のために、システム制御OSに設定されたシステム制御プログラムに対してメッセージ転送を行なう。

【0080】

前述したゲストOS間の通信では、各ゲストOSにおいて設定される各々の論理パーティション対応の論理アドレス空間を物理アドレス空間のメッセージ領域に対してマッピング、アンマッピングする処理としてメッセージ転送を実現していた。システム制御OSに設定されたシステム制御プログラムとゲストOS間の通信処理においても、図7に示すように制御OS510の生成するメッセージチャネル512を適用した通信が実行されるが、システム制御OS520は、例えばUNIX（登録商標）において通信に適用するソケットを生成してメッセージチャネルを介した通信を実現する。

【0081】

図7のように、仮想ファイルを用いた通信を行う場合は、システム制御OSは、自己の管理するファイルシステムによって識別可能なファイルディスクリプタに関連付けられたソケットを生成し、このソケットを介してアクセス可能な仮想ファイルを作成し、物理アドレス空間のメッセージ領域を仮想ファイルを介してアクセスすることで、メッセージ送受信を実行する。

【0082】

図10のように、仮想ファイルを用いない通信を行う場合は、システム制御OSは、自己の管理するファイルシステムによって識別可能なファイルディスクリプタに関連付けられたソケットを生成し、該ソケットを介して物理アドレス空間のメッセージ領域をプロセスのアドレス空間へマッピングし、プロセスは直接メッセージ領域へのアクセスを実行する。

【0083】

UNIX（登録商標）のソケットには、システム内のプロセスとの通信に使われる場合（UNIXドメインソケット）と、ネットワークを通じた別のシステムとの通信に使われる場合（インターネットドメインソケット）がある。システム制御OSは、メッセージチャネルを新たなドメインのソケットとしてシステム制御プログラムに提供する。システム制御プログラムは、既存のソケットとほぼ同じセマンティックでメッセージチャネルを用いた通信を行うことができる。

【0084】

従来のUNIX（登録商標）のソケット（UNIXドメインソケット）を適用した通信処理の概要は、以下のとおりである。

a) メッセージ受信側（サーバ）プロセスが、ソケット（`socket`）システムコールでソケットを作成する。

b) メッセージ受信側（サーバ）プロセスが、バインド（`bind`）システムコールを使用してソケットに対応する名前（ファイルディスクリプタ）を割り当てる。

c) メッセージ受信側（サーバ）プロセスが、リッスン（`listen`）システムコールを実行し、接続準備状態に設定する。

d) メッセージ送信側（クライアント）プロセスが、ソケット（`socket`）システムコールでソケットを作成し、コネクト（`connect`）システムコールを使用してサーバ側ソケットとの通信路を確立する。

e) メッセージ受信側（サーバ）プロセスが、アクセプト（`accept`）システムコールを実行し、メッセージ送信側（クライアント）プロセスからの要求を受け入れる。

f) 接続完了後に、ソケット対応のファイルディスクリプタに対する書き込み処理としてのライト（`write`）システムコールと、読み取り処理としてのリード（`read`）システムコールを利用して、メッセージ送受信を実現する。

【0085】

以上が、UNIXにおけるソケット（UNIXドメインソケット）を適用した通信シーケンスの概要である。本発明では、図7、図10に示す制御OS510の設定する通信チャンネル512を適用し、かつ、システム制御プログラム側では、上述した既存のソケットを適用した処理とほぼ同様の処理によってゲストOSとの通信を可能とする。なお、ゲストOS側の処理は、前述したゲストOS間通信における処理と同様の処理として実行される。

【0086】

図7、図8（仮想ファイルを用いた通信）および図11、図12（仮想ファイルを用いない通信）を参照してシステム制御OS520と、ゲストOS530管理の通信処理シーケンスについて説明する。なお、実際に通信を実行するのは、システム制御OS520の設定したシステム制御プログラムとゲストOSであり、通信処理には、制御OS510の生成する通信チャンネル512と、システム制御プログラムカーネル521の生成する通信用ソケットが適用されることになる。システム制御プログラムは、様々なシステムコールを実行し、システムコールに基づく処理をカーネル521が実行し、カーネルにおいて通信環境の設定、通信制御処理が実行される。システム制御プログラムは、システム制御プログラムカーネル521の生成する通信用ソケットを適用することにより、上述したUNIX対応のソケットを適用した通信とほぼ同様の処理で、ゲストOSとの通信を実行する。

【0087】

図8および図11は、ゲストOSとの通信を実行するシステム制御プログラムの処理と、カーネルの処理とを対応付けて示している。以下、各ステップ毎に処理の詳細を説明する。

【ステップS201】

まず、ステップS201において、システム制御プログラムは、接続要求を受け付けるための接続用ソケットを生成するため、ソケットシステムコール（`socket`（））を起動する。なお、システムコールはカーネル521の機能の呼出に適用される。カーネルは、ソケットシステムコール（`socket`（））の起動により、接続用ソケットを生成する。図7および図10に示す接続用ソケット525である。

なお、生成されるソケットには、システム制御OS520において管理するファイルシステムに設定されるファイル名（ファイルディスクリプタ）が対応付けられる。

【0088】

【ステップS202】

次に、システム制御プログラムは、ステップS202において、バインドシステムコール（`bind`（`socket`，・・・））を起動する。カーネル521は、バインドシステムコール（`bind`（`socket`，・・・））に基づいて、システム制御プログラム接続

女ホを付したののソールへ1レ（ホード田カ）とソソソドを対応付ける。ソソソドは、システム制御プログラムの提供するサービスに対応付けられて設定されることになる。

【0089】

【ステップS203】

次に、システム制御プログラムは、ステップS203において、リッスンシステムコール（listen（socket，・・・））を起動する。カーネルは、リッスンシステムコール（listen（socket，・・・））に基づいて、ソケットに対応付けたサービスの接続を許可する。ソケットには、対応するサービス識別子が設定されており、システム制御プログラムは、サービス対応の通信許可設定を実行することができる。

【0090】

【ステップS204】

次に、システム制御プログラムは、ステップS204において、アクセプトシステムコール（fd=accept（socket，・・・））を起動する。カーネル521は、アクセプトシステムコール（fd=accept（socket，・・・））に基づいて、メッセージチャネルの接続要求の有無を検証し、接続メッセージチャネルの接続要求があるかどうか調べる。

接続要求が到着していたら以下の（a）～（c）の処理を実行する。

（a）通信用のソケット（図7および図10の通信用ソケット526）を作成する。

（b）システム制御OS520の管理するファイルシステムのファイル記述子（ファイルディスクリプタ：fd）テーブル（fd-table）にソケット対応のファイル記述子（ファイルディスクリプタ：fd）を登録する。

（c）登録されたソケット対応のファイル記述子（ファイルディスクリプタ：fd）をシステム制御プログラムに返す。

接続要求が到着していなかったら以下の（d），（e）の処理を実行する。

（d）プロセスをサスペンド状態にする。

（e）次に接続要求が到着したときにプロセスのサスペンド状態を解除して、上記の（a）～（c）の処理を行う。

【0091】

以上の処理によってシステム制御プログラムとゲストOS間にメッセージチャネルが設定される。図7に示す仮想ファイル524は、制御OS510の管理する物理アドレス空間メッセージ領域511を抽象化したファイルである。ファイルの作成処理およびファイルの削除処理は、システム制御OS520によって実行され、ゲストOS530からシステム制御プログラムに対するメッセージ転送時において、メッセージ転送前は、ゲストOS530の論理アドレス空間メッセージ領域533が物理アドレス空間メッセージ領域511にマッピングされた状態にあり、メッセージ転送後は、システム制御OS520側の仮想ファイル524として作成され、システム制御プログラムからアクセス可能な状態になる。また、システム制御プログラムからゲストOS530に対するメッセージ転送時においては、メッセージ転送前は、システム制御OS520側の仮想ファイル524がシステム制御プログラムからアクセス可能な状態にあり、メッセージ転送後は、システム制御OS520が仮想ファイルを削除し、物理アドレス空間メッセージ領域511がゲストOS530の論理アドレス空間メッセージ領域533にマッピングされる。

【0092】

また、物理アドレス空間メッセージ領域511を仮想ファイル524を介するのではなく、直接プロセスのアドレス空間にマップする場合は、以下のように処理される。ゲストOS530からシステム制御プログラムに対するメッセージ転送時において、メッセージ転送前は、ゲストOS530の論理アドレス空間メッセージ領域533が物理アドレス空間メッセージ領域511にマッピングされた状態にあり、メッセージ転送後は、システム制御OS520が該物理アドレス空間メッセージ領域511をプロセスのアドレス空間にマップして、システム制御プログラムから直接アクセス可能な状態にする。また、システム制御プログラムからゲストOS530に対するメッセージ転送時においては、メッセー

プロセスのアドレス空間にマップして、システム制御プログラムから直接アクセス可能な状態にあり、メッセージ転送後は、システム制御OS520が該マッピングを解除し、物理アドレス空間メッセージ領域511がゲストOS530の論理アドレス空間メッセージ領域533にマッピングされる。

【0093】

なお、仮想ファイルを用いる場合、システム制御プログラムは、通信用ソケットのファイルディスクリプタを指定したリードシステムコールによって、該ソケット対応の仮想ファイル524の識別子を取得することができる。システム制御プログラムは、この仮想ファイルIDを指定したオープン（open）、クローズ（close）、リード（read）、ライト（write）のいずれかのシステムコールを適用して、ソケット対応のファイルのオープン（open）、クローズ（close）、さらにメッセージの書き込み【ライト（write）】、読み取り【リード（read）】が実行可能となる。

【0094】

また、物理アドレス空間メッセージ領域511を仮想ファイル524を介するのではなく、直接プロセスのアドレス空間にマップする場合は、システム制御プログラムは、通信用ソケットのファイルディスクリプタを指定したリードシステムコールによって、物理アドレス空間メッセージ領域511がプロセスのアドレス空間にマップされたアドレスを取得することができる。システム制御プログラムは、このアドレスに対する直接のアクセスを行うことにより、メッセージ領域の読み取りおよび書き込みが実行可能となる。

【0095】

システム制御プログラムは、複数のソケットを設定することが可能であり、それぞれのソケットに対応して個別の仮想ファイルが設定され、それぞれの仮想ファイルは、個別に物理アドレス空間のメッセージ領域に対応付けられ個別の通信が可能となる（仮想ファイルではなく物理アドレス空間メッセージ領域を直接プロセスのアドレス空間にマップする場合も同様）。それぞれのソケットにはサービス識別子に対応付けられており、ソケット指定のシステムコールの選択的実行により、通信許可、不許可の設定など、ソケット（サービス）毎の個別の処理が実行できる。

【0096】

図8および図11に示すステップS-A～Eの各処理は、ステップS201～S204の処理によるメッセージチャネルの生成後に任意のタイミングで実行可能な処理である。各処理について説明する。

【0097】

【ステップS-A】

ステップS-Aは、ゲストOSからのメッセージ受信処理である。メッセージ受信に際して、システム制御プログラムは、リードシステムコール（read（fd，・・・））を起動する。カーネル521は、リードシステムコール（read（fd，・・・））に基づいて、ゲストOSからのメッセージチャネルを介したメッセージ到着の有無を検証する。

仮想ファイルを用いた通信を行う場合は、メッセージが到着していたら以下の（a）、（b）の処理を実行する。

（a）到着したメッセージに対応する仮想ファイルを作成する。図7に示す仮想ファイル524である。

（b）到着したメッセージ領域に対応する仮想ファイルの識別子（ID）とショートメッセージをユーザバッファにコピーする。すなわちシステム制御プログラムに通知する。システム制御プログラムは、この仮想ファイルの識別子（ID）に基づいて仮想ファイルを指定してメッセージを読み取ることができる。

メッセージが到着していなかったら、以下の（c）、（d）の処理を実行する。

（c）プロセスをサスペンド状態にする。

（d）次にメッセージが到着したときにプロセスのサスペンド状態を解除して、上記（a）、（b）の処理を行う。

図 7 に示す仮想ファイル 5 2 4 は、ソケット対応のファイル記述子（ファイルディスクリプタ：f d）によって特定されるファイルであり、制御 OS 5 1 0 の管理する物理アドレス空間メッセージ領域 5 1 1 に対応可能なファイルである。

【 0 0 9 9 】

仮想ファイルを用いない通信（物理アドレス空間メッセージ領域を直接プロセスのアドレス空間にマップする場合）では、メッセージ領域をプロセスのアドレス空間にマップし、そのアドレスとサイズをユーザバッファにコピーする。すなわちシステム制御プログラムに通知する。システム制御プログラムはこのアドレスに基づいて、メッセージ領域の内容を直接読み取ることができる。

【 0 1 0 0 】

【ステップ S-B】

ステップ S-B は、メッセージに対するアクセスに関する処理である。

仮想ファイルを用いた通信を行う場合は、すなわち、仮想ファイル 5 2 4 に対する処理である。システム制御プログラムは、仮想ファイル 5 2 4 に対する処理として、オープン（open）、クローズ（close）、リード（read）、ライト（write）のいずれかのシステムコールを起動することができる。カーネル 5 2 1 は、各システムコールに対応する処理を実行する。オープン（open）、クローズ（close）は、仮想ファイル 5 2 4 のオープンとクローズ処理に対応し、ライト（write）は、仮想ファイル 5 2 4 に対するデータ書き込み、リード（read）は、仮想ファイル 5 2 4 からのデータ読み取りに相当する。これらの処理は、ファイル記述子を指定した処理として実行される。

【 0 1 0 1 】

仮想ファイルを用いない通信（物理アドレス空間メッセージ領域を直接プロセスのアドレス空間にマップする場合）では、システム制御プログラムは、プロセスのアドレス空間にマップされたメッセージ領域を直接アクセスすることができる。

【 0 1 0 2 】

【ステップ S-C】

ステップ S-C は、メッセージの送信の際の処理である。

仮想ファイルを用いた通信の場合は、仮想ファイル 5 2 4 に書き込まれたメッセージの送信処理である。システム制御プログラムが、通信用ソケットに対して仮想ファイルの I D を指定したライトシステムコール（write（fd，・・・））を起動することで、カーネル 5 2 1 は、ライトシステムコール（write（fd，・・・））に基づいて、仮想ファイル 5 2 4 に書き込まれたメッセージの送信処理を実行する。

【 0 1 0 3 】

制御 OS 5 1 0 は、カーネル 5 2 1 が、ライトシステムコール（write（fd，・・・））に基づいて実行する処理に応じて、物理アドレス空間メッセージ領域 5 1 1 のシステム制御 OS 5 2 0 側の仮想ファイル 5 2 4 を削除し、ゲスト OS 5 3 0 側の論理アドレス空間メッセージ領域 5 3 3 にマッピングする処理を実行する。この処理により、システム制御プログラムが仮想ファイル 5 2 4 を適用して書き込みを実行したメッセージが、ゲスト OS 5 3 0 に転送されることになる。ゲスト OS 5 3 0 側の処理は、前述したと同様、メッセージポート 5 3 2、イベントポート 5 3 1 を適用した処理として実行される。

【 0 1 0 4 】

仮想ファイルを用いない通信（物理アドレス空間メッセージ領域を直接プロセスのアドレス空間にマップする場合）では、システム制御プログラムが、通信用ソケットに対して物理アドレス空間メッセージ領域がプロセスのアドレス空間にマップされているアドレスを指定したライトシステムコール（write（fd，・・・））を起動することで、カーネル 5 2 1 は、ライトシステムコール（write（fd，・・・））に基づいて、物理アドレス空間メッセージ領域の送信処理を実行する。メッセージ領域のプロセスアドレス空間へのマップは、送信時に解除される。

【0105】

【ステップS-D】

ステップS-Dは、受信メッセージを削除する際の処理である。

仮想ファイルを用いた通信の場合、システム制御プログラムが、仮想ファイル524に対してアンリンクシステムコール（`unlink（）`）を起動することで、カーネル521は、アンリンクシステムコール（`unlink（）`）に基づいて、受信メッセージを特定するファイルの削除によりメッセージの破棄を実行する。

仮想ファイルを用いない通信では、単にプロセスのアドレス空間へのマップを解除し、物理アドレス空間メッセージ領域を開放する。

【0106】

【ステップS-E】

ステップS-Eは、メッセージチャネルの切断処理であり、システム制御プログラムが、ソケットを指定したクローズシステムコール（`close（socket）`）を起動することで、カーネル521は、メッセージポートを削除し、メッセージチャネルの切断を行なう。

【0107】

なお、図8、図11には示していないが、システム制御プログラムはセレクトシステムコール（`select`）によって、メッセージの送受信処理の監視が実行できる。ソケットに対応するファイルディスクリプタを特定したセレクトシステムコールを起動することで、カーネルは、セレクトシステムコール（`select`）によって特定されるソケットに対する受信メッセージの有無を他の入出力処理と同様、常に監視することが可能となる。たとえば、システム制御プログラムが複数のゲストOSとそれぞれ異なる通信ソケットを設定した通信を行なう場合に、ゲストOS（A）との通信ソケット【`fd=s0`】と、ゲストOS（B）の通信ソケット【`fd=s1`】をセレクトシステムコールの対象として指定することで、カーネルは、各ソケットに対するメッセージの有無を監視し、メッセージが有る場合にシステム制御プログラムに通知する処理を行なう。このように、セレクトシステムコール（`select`）の利用により、効率的な処理が可能となる。

【0108】

次に、図9および図12に示すフローチャートを参照して、システム制御プログラムがゲストOS側からのメッセージを受信し、その後システム制御プログラムからゲストOSに対してメッセージを送信する処理を実行する場合における、システム制御プログラムおよびゲストOSの処理について説明する。

【0109】

まず、ステップS301において、システム制御プログラムはソケットシステムコール（`socket（）`）でソケットを作成する。このソケットは、図7に示す接続用ソケット525に相当する。

ステップS302において、システム制御プログラムはバインドシステムコール（`bind（）`）でソケットにサービスIDを付与する。前述したように、各ソケットは、システム制御プログラムの提供するサービスに対応付けられて設定されることになる。

【0110】

ステップS303で、システム制御プログラムはリスンシステムコール（`listen（）`）でソケットを接続許可状態にする。システム制御プログラムは、ソケットを特定したリスンシステムコールの発行によって、ソケット毎に接続許可状態、不許可状態の設定をすることが可能となる。

【0111】

ステップS304で、システム制御プログラムはアクセプトシステムコール（`accept（）`）でソケットに対する接続要求を待つ。

【0112】

次に、ステップS305で、ゲストOSがシステム制御プログラムに対してメッセージチャネルを接続する操作を行う。

この処理には、以下の処理が含まれる。

a) ゲストOSからの接続要求処理（図6のシーケンス図におけるステップS101の接続要求処理に相当）

b) 制御OSが接続要求をシステム制御プログラムに通知し、先に説明したステップS304のアクセプトシステムコールが発行されたことに基づいて、制御OSによるメッセージチャネル設定通知をゲストOSに通知する処理（図6に示すステップS104に相当）

以上の処理が含まれる。

【0113】

ステップS306で、システム制御プログラム側のカーネルによってメッセージチャネルの接続が行われると同時に、カーネルは、システム制御プログラムに対するソケットの接続処理を行う。すなわち、通信用のソケットを作成し、アクセプトシステムコール（accept（））での待ち状態から復帰する。

【0114】

ステップS307で、ゲストOSはメッセージ領域を用意し、メッセージチャネルを通じて転送する。この処理には、以下の処理が含まれる。

a) ゲストOSによるメッセージ領域確保処理、すなわち、ゲストOSに対応する論理パーティションの論理アドレス空間にメッセージ領域を設定する処理

b) ゲストOSの論理アドレス空間のメッセージ領域を物理アドレス空間のメッセージ領域にマッピングする処理

以上の処理が含まれる。

【0115】

仮想ファイルを用いた通信を行う場合は、ステップS308（a）で、システム制御プログラム側のカーネルは、ゲストOSから送られたメッセージ領域を受信し、システム制御プログラムがアクセス可能な仮想ファイルを作成する。図7に示す仮想ファイル524である。なお、作成される仮想ファイルには仮想ファイルIDが設定される。なお、制御OSは、システム制御プログラム側カーネルの生成した仮想ファイル524に物理アドレス空間のメッセージ領域を対応付ける処理を実行する。

【0116】

仮想ファイルを用いない通信を行う場合は、ステップS308（b）で、システム制御プログラム側のカーネルは、ゲストOSから送られたメッセージ領域を受信し、システム制御プログラムのアドレス空間にマップする。

【0117】

仮想ファイルを用いた通信を行う場合は、ステップS309（a）で、システム制御プログラムが通信用ソケットを指定したリードシステムコール（read（））によって、メッセージ領域に対応する仮想ファイルのIDを取得する。この仮想ファイルIDは、例えばファイル名である。

【0118】

ステップS310（a）で、システム制御プログラムは仮想ファイルを、仮想ファイルIDを指定したオープンシステムコール（open（））によりオープンし、さらに、リードシステムコール（read（））により、メッセージ領域に格納されたデータを読み取る。

【0119】

さらに、システム制御プログラムは、オープンした仮想ファイルに対してライトシステムコール（write（））を適用して、仮想ファイルにデータを書き込むことができる。

【0120】

仮想ファイルを用いない通信を行う場合は、システム制御プログラムは、自らのアドレス空間にマップされたメッセージ領域を直接アクセス（ステップS309（b）、S310（b））すればよい。

ステップ S 3 1 1 (a) またはステップ S 3 1 1 (b) において、システム制御プログラムが通信用ソケットを指定して、仮想ファイルの I D、もしくは物理アドレス空間メッセージ領域がマップされたプロセスアドレス空間のアドレスの書き込み処理としてのライトシステムコール (w r i t e ()) を実行すると、カーネルはメッセージ領域をゲスト O S に転送する処理を行う。この処理に応じて、制御 O S は、システム制御プログラム側の仮想ファイルの削除、もしくはプロセスのアドレス空間にマッピングされていた物理アドレス空間のメッセージ領域をアンマップするとともに、ゲスト O S の論理パーティションのメッセージ領域にマッピングする。この処理により、メッセージがゲスト O S によって読み取り可能となり、ゲスト O S によるメッセージ読み取りが実行される。

【 0 1 2 2 】

上述したように、システム制御プログラムとゲスト O S 間の通信処理においては、システム制御プログラム側は、ソケットを適用した通信プロセスを実行することで制御 O S の設定する通信チャネルを介したゲスト O S との通信が可能となる。

【 0 1 2 3 】

以上、特定の実施例を参照しながら、本発明について詳解してきた。しかしながら、本発明の要旨を逸脱しない範囲で当業者が該実施例の修正や代用を成し得ることは自明である。すなわち、例示という形態で本発明を開示してきたのであり、限定的に解釈されるべきではない。本発明の要旨を判断するためには、特許請求の範囲の欄を参酌すべきである。

【 0 1 2 4 】

なお、明細書中において説明した一連の処理はハードウェア、またはソフトウェア、あるいは両者の複合構成によって実行することが可能である。ソフトウェアによる処理を実行する場合は、処理シーケンスを記録したプログラムを、専用のハードウェアに組み込まれたコンピュータ内のメモリにインストールして実行させるか、あるいは、各種処理が実行可能な汎用コンピュータにプログラムをインストールして実行させることが可能である。

【 0 1 2 5 】

例えば、プログラムは記録媒体としてのハードディスクや R O M (Read Only Memory) に予め記録しておくことができる。あるいは、プログラムはフレキシブルディスク、C D - R O M (Compact Disc Read Only Memory)、M O (Magneto optical) ディスク、D V D (Digital Versatile Disc)、磁気ディスク、半導体メモリなどのリムーバブル記録媒体に、一時的あるいは永続的に格納 (記録) しておくことができる。このようなリムーバブル記録媒体は、いわゆるパッケージソフトウェアとして提供することができる。

【 0 1 2 6 】

なお、プログラムは、上述したようなリムーバブル記録媒体からコンピュータにインストールする他、ダウンロードサイトから、コンピュータに無線転送したり、L A N (Local Area Network)、インターネットといったネットワークを介して、コンピュータに有線で転送し、コンピュータでは、そのようにして転送されてくるプログラムを受信し、内蔵するハードディスク等の記録媒体にインストールすることができる。

【 0 1 2 7 】

なお、明細書に記載された各種の処理は、記載に従って時系列に実行されるのみならず、処理を実行する装置の処理能力あるいは必要に応じて並列的にあるいは個別に実行されてもよい。また、本明細書においてシステムとは、複数の装置の論理的集合構成であり、各構成の装置が同一筐体内にあるものには限らない。

【産業上の利用可能性】

【 0 1 2 8 】

以上、説明したように、本発明の構成によれば、複数のオペレーティングシステム (O S) が共存するマルチ O S 環境における O S 間通信において、O S 間の通信制御を実行する制御 O S を設定し、制御 O S が、通信実行 O S 各々に対応して設定される論理パーティ

ンション相互のプロセッサ転送制御を大行する構成とし、物理アドレス空間のプロセッサ領域を、メッセージ送信側OSの論理パーティションアドレス空間のメッセージ領域へのマッピング状態から、メッセージ受信側OSの論理パーティションアドレス空間のメッセージ領域へのマッピング状態に切り替える処理を実行してOS間のメッセージ転送制御を行なう構成としたので、異なるOS間の通信におけるスムーズな通信が可能となる。

【0129】

さらに、本発明の構成によれば、通信実行OSが、自己の管理するファイルシステムによって識別可能なファイルディスクリプタに関連付けられたソケットを生成し、ソケットを介してアクセス可能な仮想ファイルを設定し、仮想ファイルを介してメッセージの送受信を実行する、または、物理アドレス空間のメッセージ領域をプロセスのアドレス空間にマッピングしてメッセージ送受信を実行する構成としたので、異なるOSとの通信において汎用的なソケットを適用したメッセージパッシングが実現される。

【0130】

さらに、本発明の構成によれば、ソケットに対応するサービス識別子を設定することにより、サービス対応の通信許可設定が可能となり、また、例えばシステムコールとしてのセレクトシステムコールの適用により、ソケットを介したメッセージの受信監視処理が可能となる。

【図面の簡単な説明】

【0131】

【図1】 本発明の情報処理装置の構成例を示す図である。

【図2】 プロセッサモジュールの構成例を示す図である。

【図3】 本発明の情報処理装置のオペレーションシステム構成を説明する図である。

【図4】 ゲストOS間の通信処理について説明する図である。

【図5】 メッセージチャネルを介した通信におけるアドレスマッピングについて説明する図である。

【図6】 ゲストOS間の通信処理シーケンスについて説明する図である。

【図7】 仮想ファイルを用いた通信の場合の、システム制御OSと、ゲストOS間の通信処理について説明する図である。

【図8】 仮想ファイルを用いた通信の場合の、システム制御OSと、ゲストOS間の通信処理において実行されるシステムコールに基づく処理について説明する図である。

【図9】 仮想ファイルを用いた通信の場合の、システム制御OSと、ゲストOS間の通信処理について説明するフロー図である。

【図10】 仮想ファイルを用いない通信の場合の、システム制御OSと、ゲストOS間の通信処理について説明する図である。

【図11】 仮想ファイルを用いない通信の場合の、システム制御OSと、ゲストOS間の通信処理において実行されるシステムコールに基づく処理について説明する図である。

【図12】 仮想ファイルを用いない通信の場合の、システム制御OSと、ゲストOS間の通信処理について説明するフロー図である。

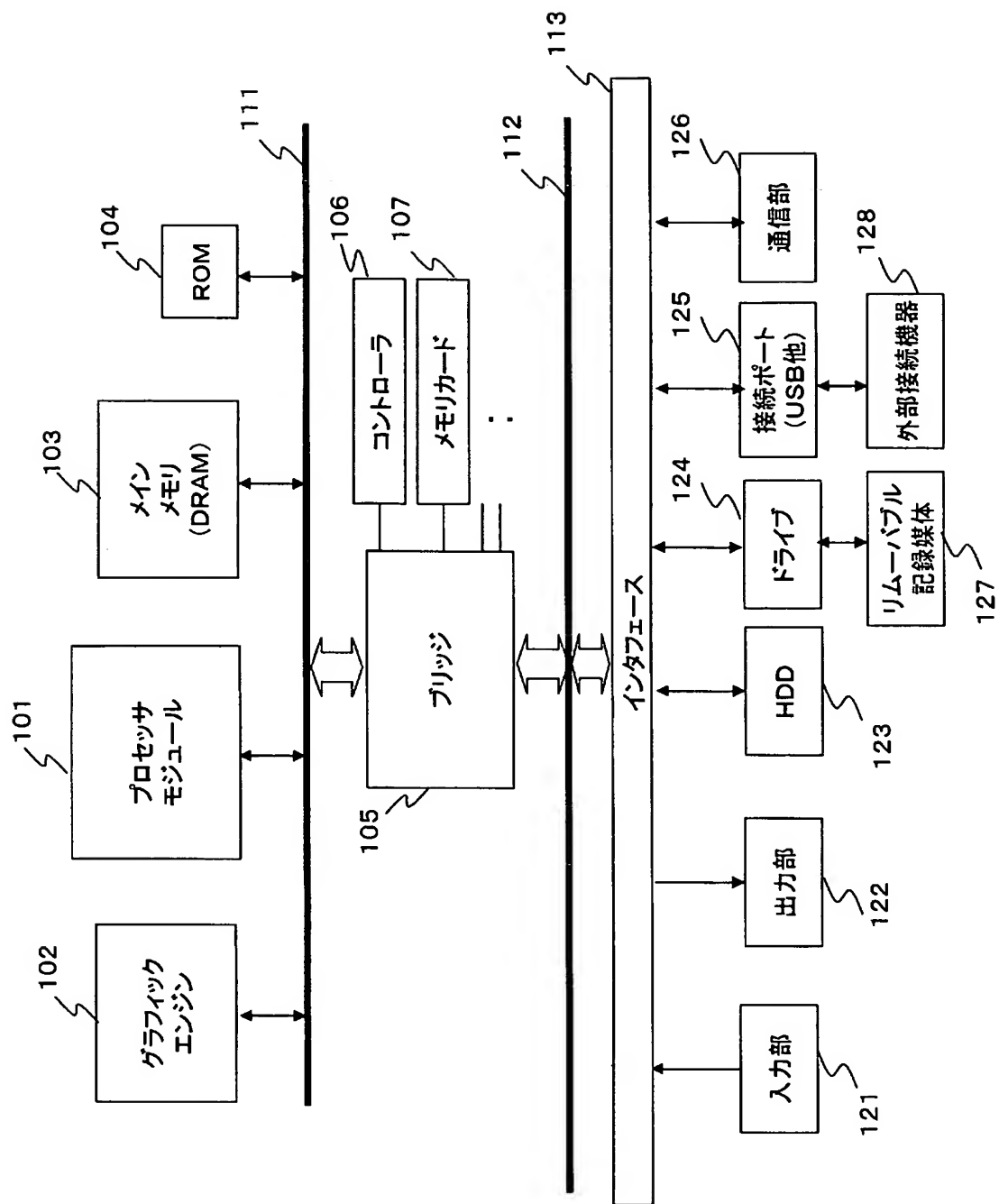
【符号の説明】

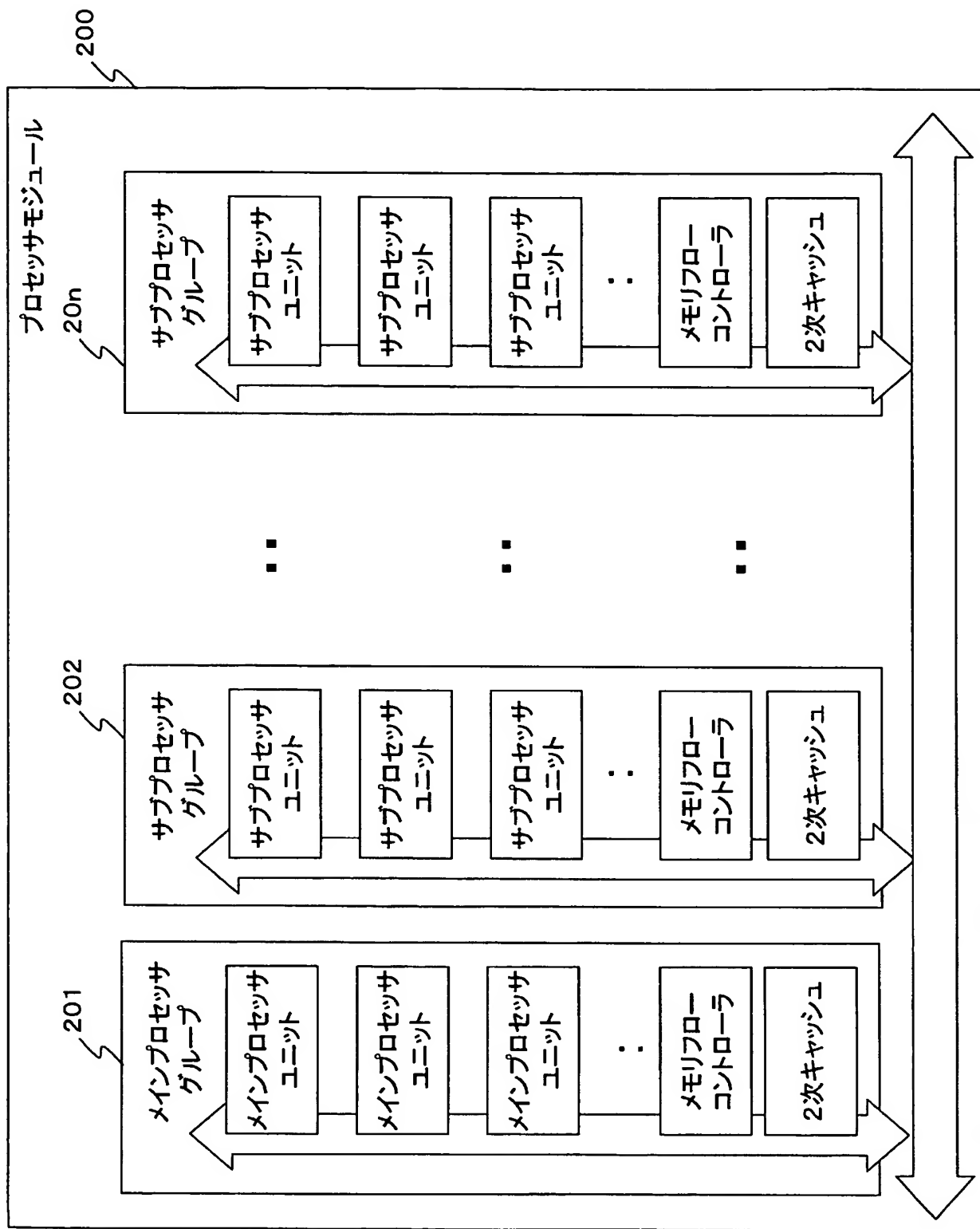
【0132】

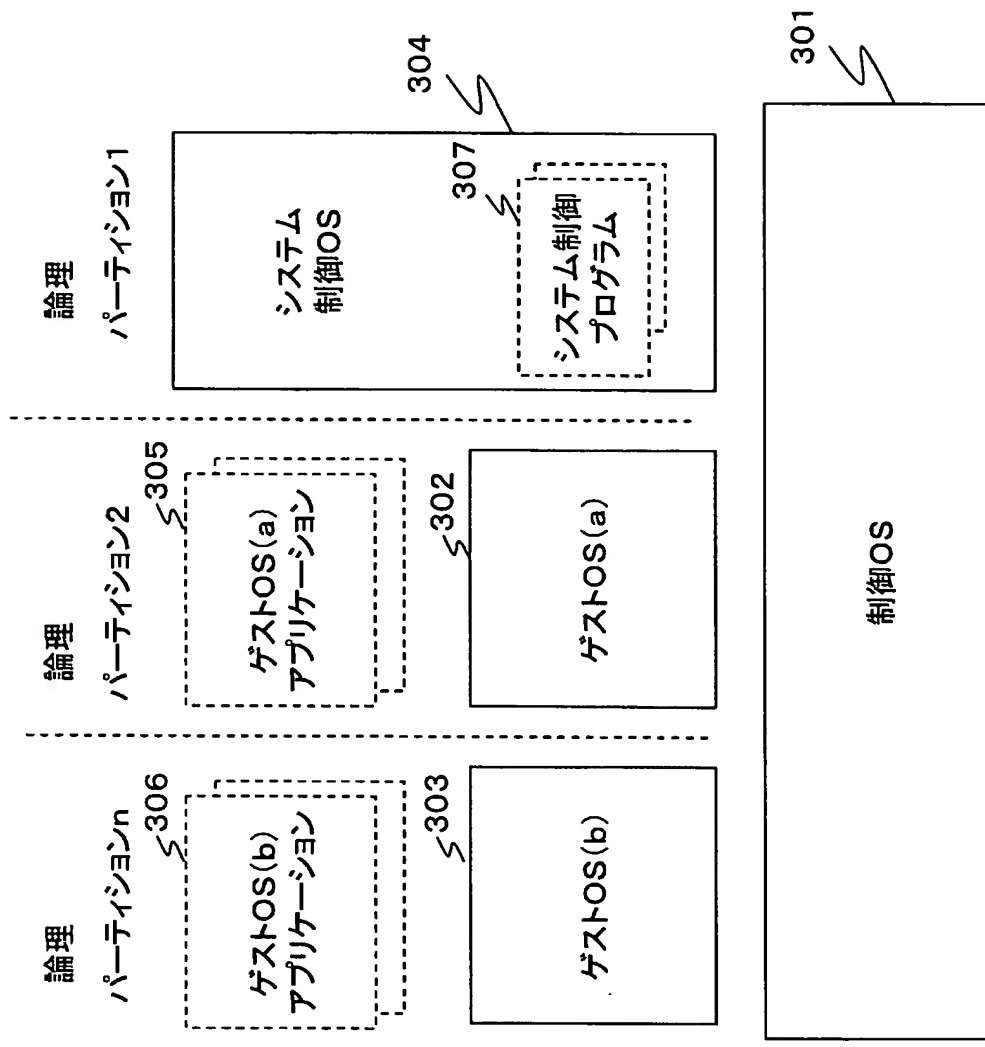
- 101 プロセッサモジュール
- 102 グラフィックエンジン
- 103 メインメモリ (DRAM)
- 104 ROM
- 105 ブリッジ
- 106 コントローラ
- 107 メモリカード
- 111 ホストバス

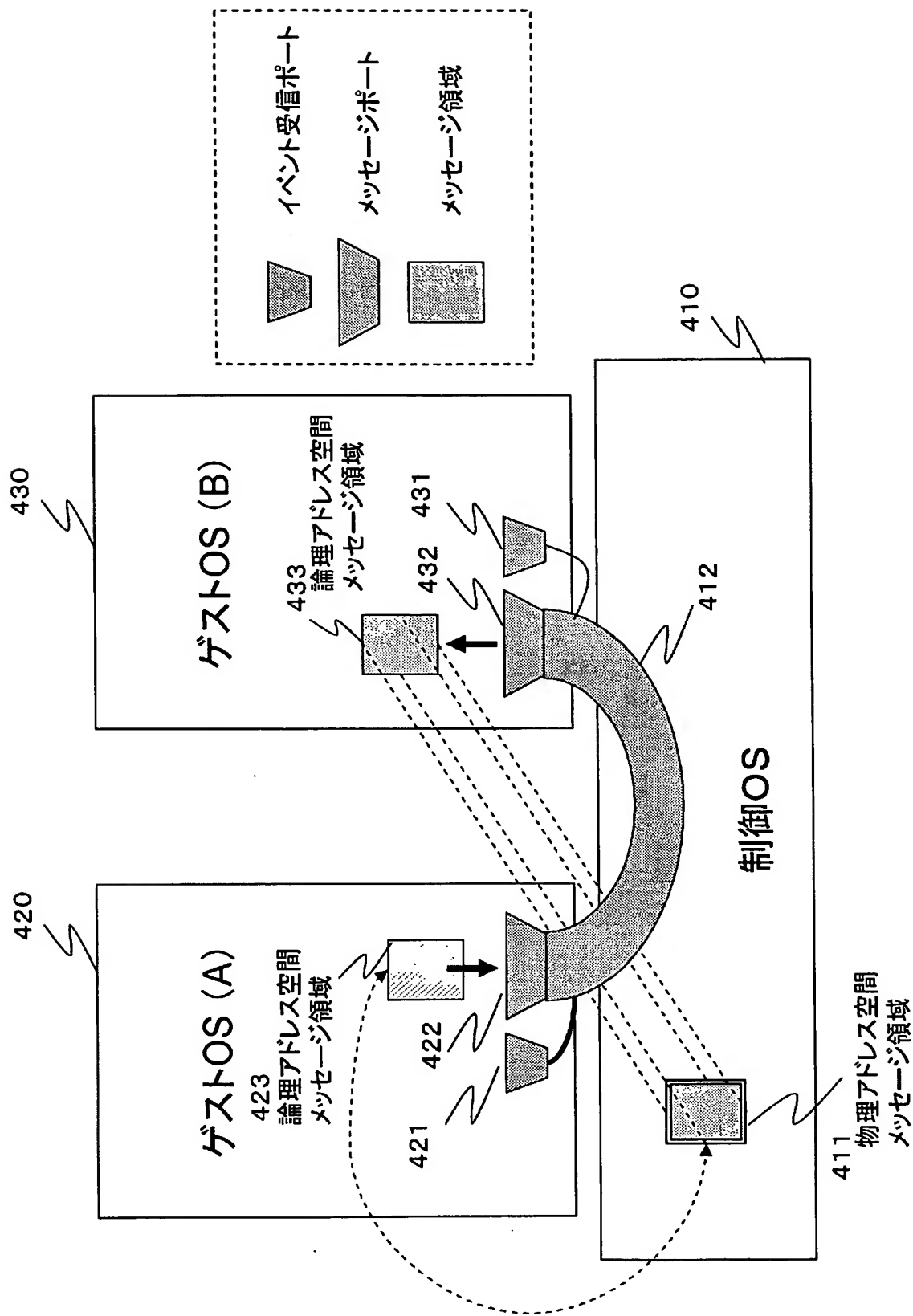
1 1 2 ハードウェア
1 1 3 インタフェース
1 2 1 入力部
1 2 2 出力部
1 2 3 HDD
1 2 4 ドライブ
1 2 5 接続ポート
1 2 6 通信部
1 2 7 リムーバブル記録媒体
1 2 8 外部接続機器
2 0 0 プロセッサモジュール
2 0 1 メインプロセッサグループ
2 0 2 ~ 2 0 n サブプロセッサグループ
3 0 1 制御OS
3 0 2 , 3 0 3 ゲストOS
3 0 4 システム制御OS
3 0 5 , 3 0 6 ゲストOSアプリケーション
3 0 7 システム制御プログラム
4 1 0 制御OS
4 1 1 物理アドレス空間メッセージ領域
4 1 2 メッセージチャネル
4 2 0 ゲストOS (A)
4 2 1 イベント受信ポート
4 2 2 メッセージポート
4 2 3 論理アドレス空間メッセージ領域
4 3 0 ゲストOS (B)
4 3 1 イベント受信ポート
4 3 2 メッセージポート
4 3 3 論理アドレス空間メッセージ領域
4 5 0 物理アドレス空間
4 5 1 メッセージ領域
4 6 0 ゲストOS (A) 論理パーティションアドレス空間
4 6 1 メッセージ領域
4 7 0 ゲストOS (B) 論理パーティションアドレス空間
4 7 1 メッセージ領域
5 1 0 制御OS
5 1 1 物理アドレス空間メッセージ領域
5 1 2 メッセージチャネル
5 2 0 システム制御OS
5 2 1 カーネル
5 2 4 仮想ファイル
5 2 5 接続用ソケット
5 2 6 通信用ソケット
5 2 7 プロセスアドレス空間へのマップ
5 3 0 ゲストOS (A)
5 3 1 イベント受信ポート
5 3 2 メッセージポート
5 3 3 論理アドレス空間メッセージ領域

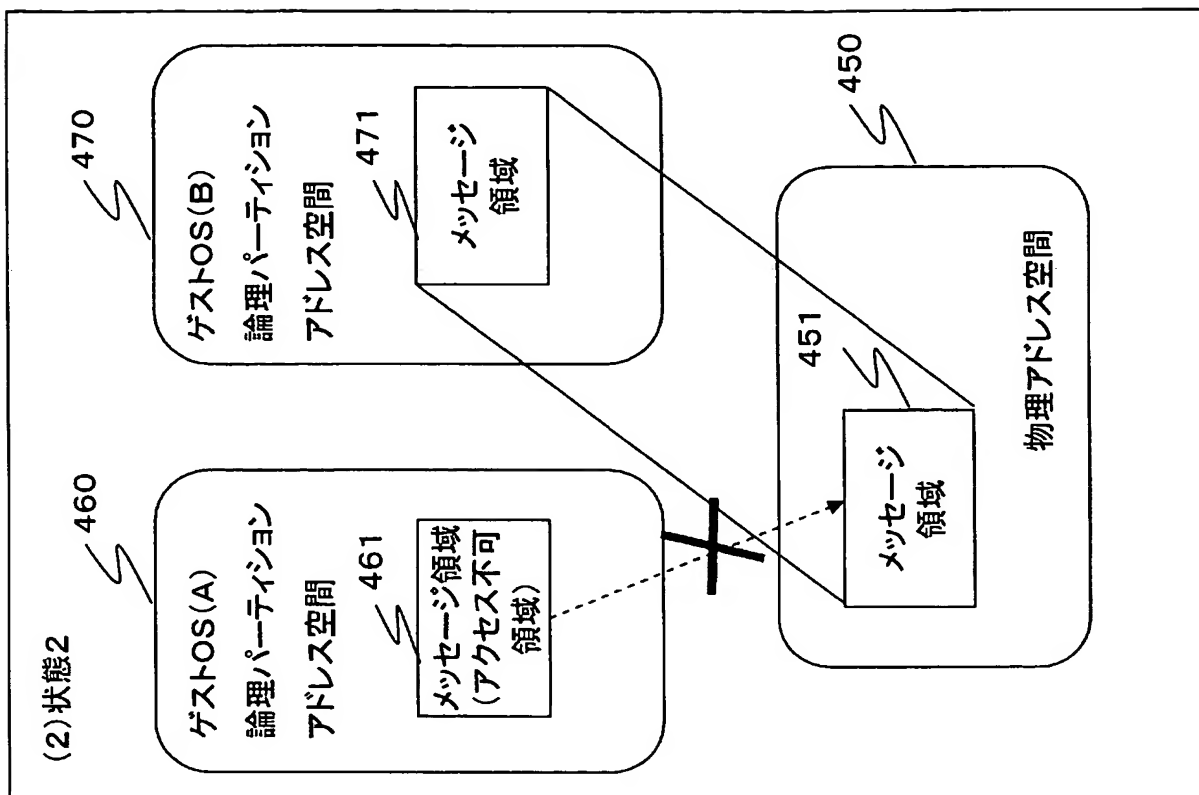
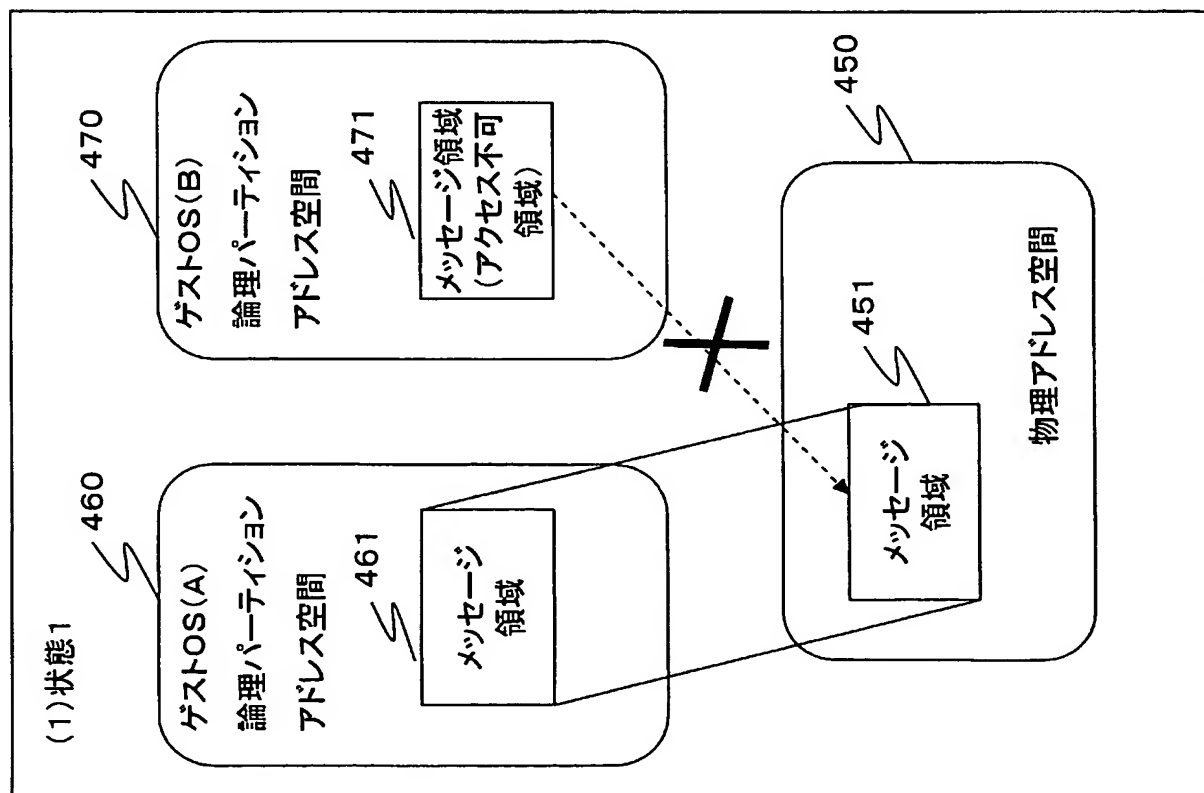
【図 1】







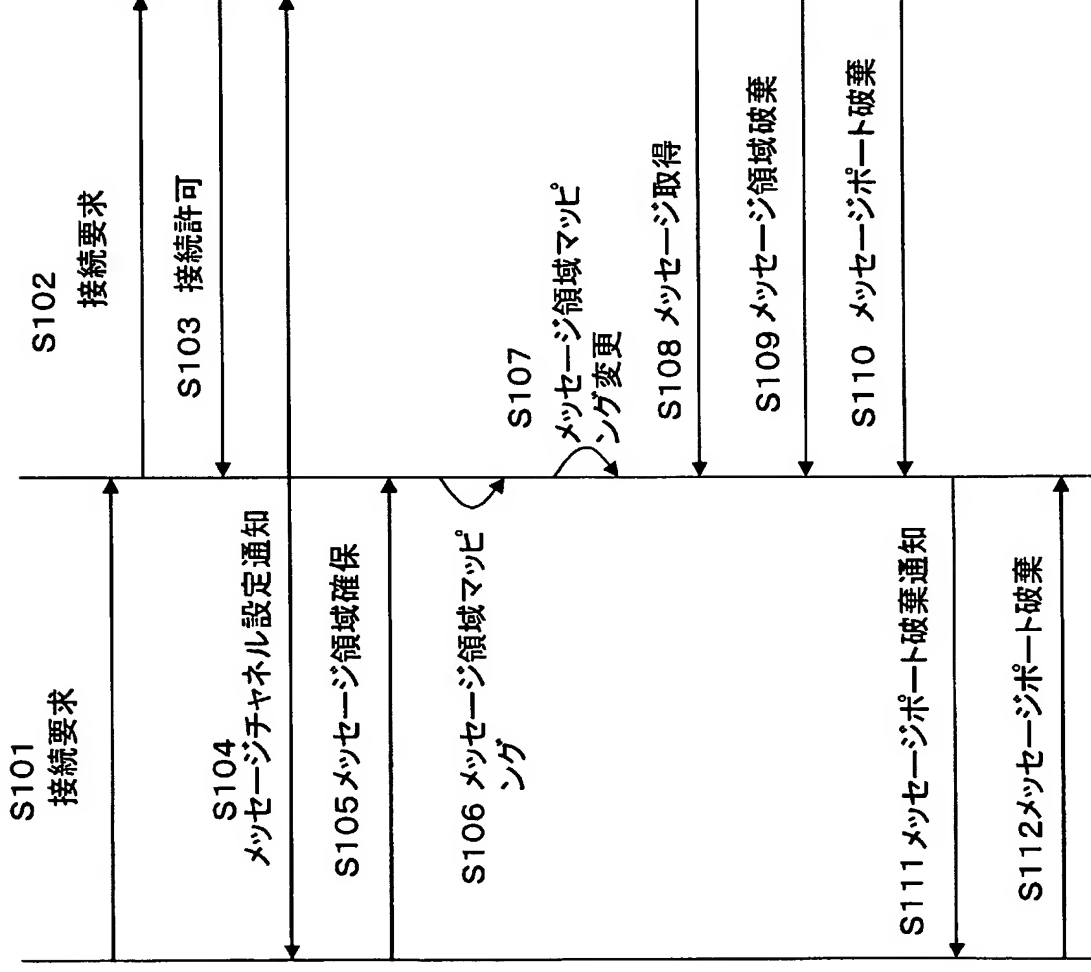


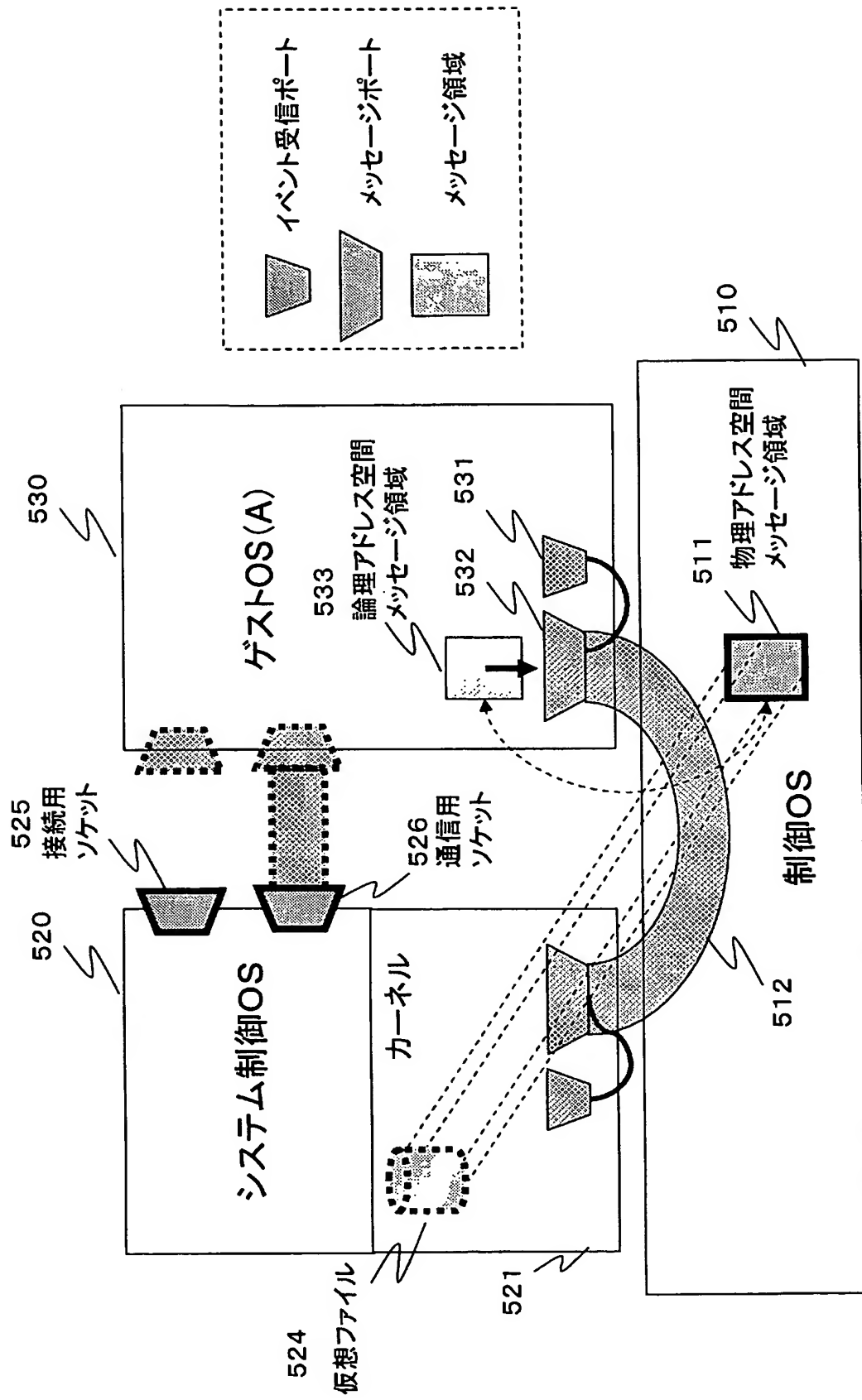


メッセージ送信側OS
(クライアント)

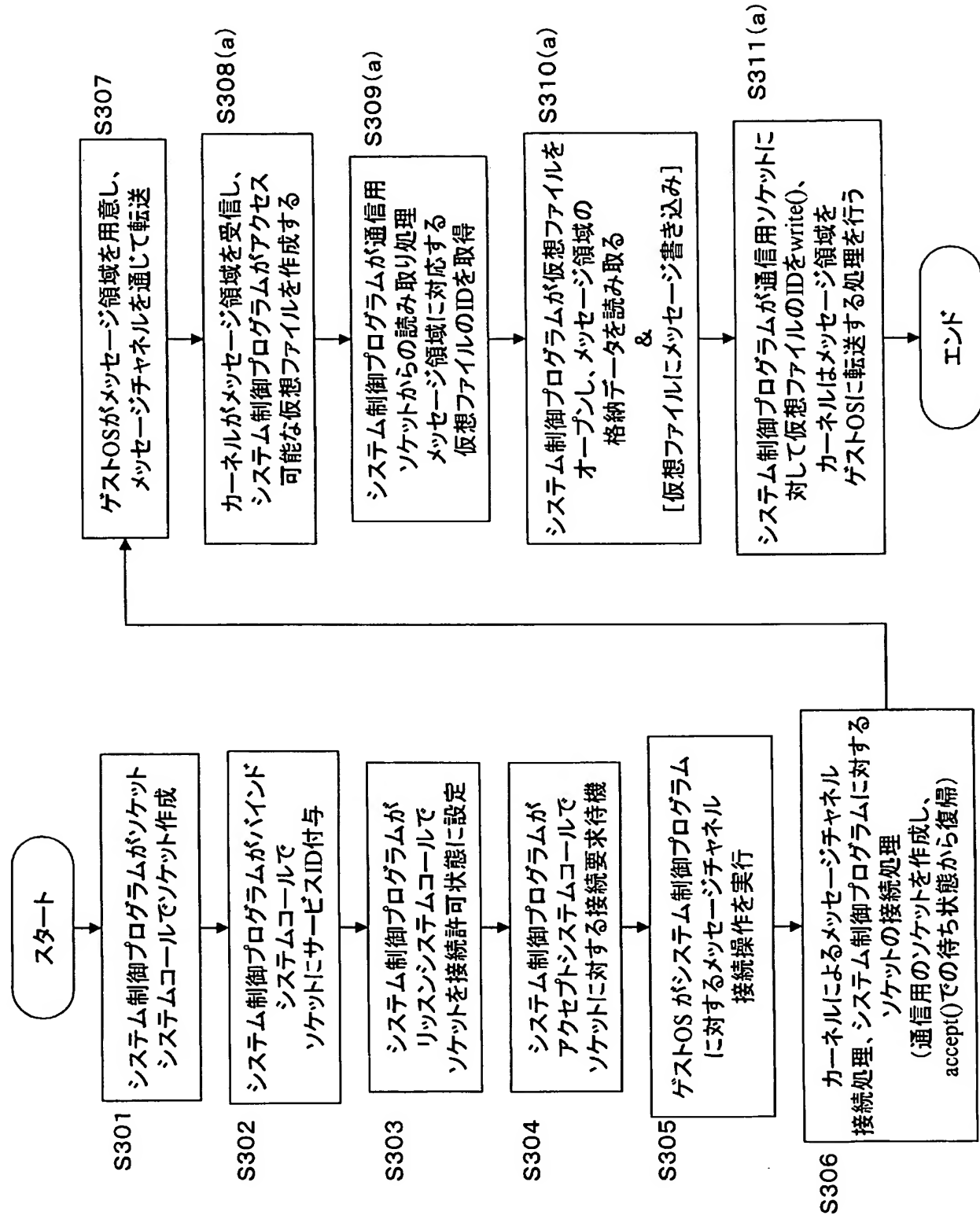
メッセージ受信側OS
(サーバ)

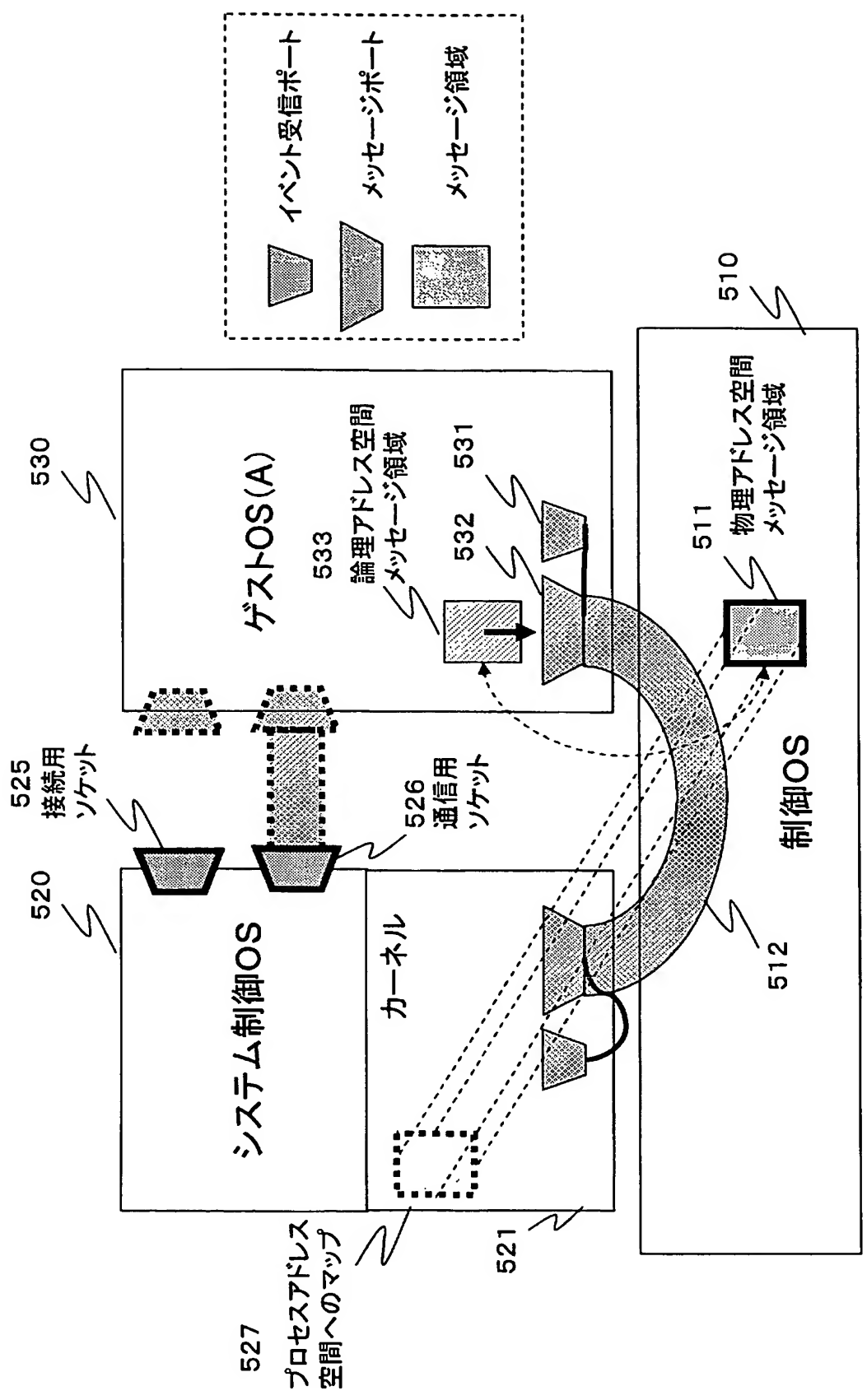
制御OS



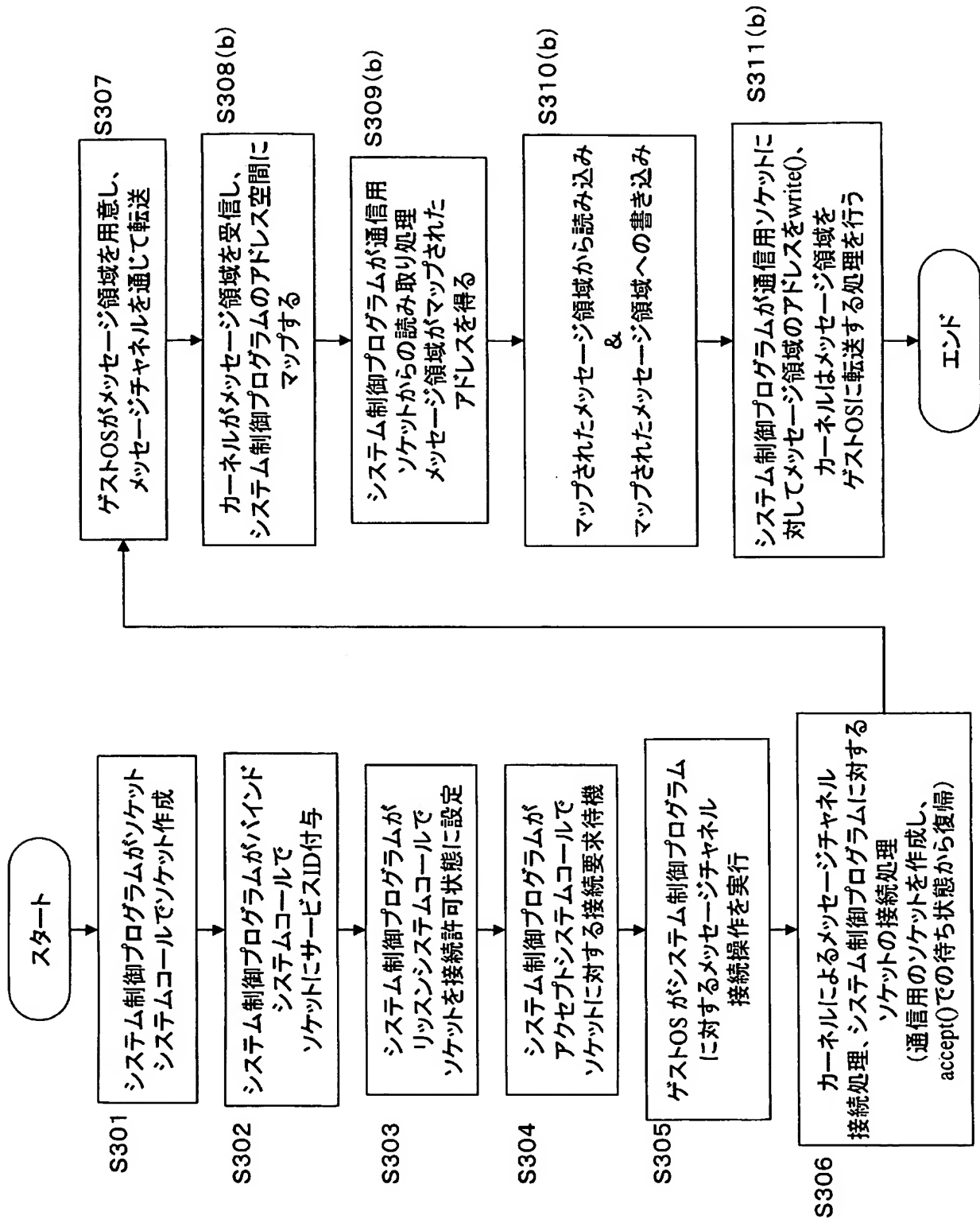


	システム制御プログラムでの 処理	対応するカーネル内部の処理(メッセージチャネルインタフェース)
S201	socket = socket(...)	ソケットを作成。メッセージチャネルの接続要求を受けるための準備をする。
S202	bind(socket, ...)	システム制御プログラムが接続要求を待つためのサービスIDとソケットを対応づける。
S203	listen(socket, ...)	指定したサービスへの接続を許可する。
S204	fd = accept(socket, ...)	メッセージチャネルを接続する。
S-A (a)	read(fd, ...)	メッセージを受信して対応するファイル名を得る。
S-B (a)	メッセージファイルを open/close/read/write	メッセージの内容を読み書き。
S-C (a)	write(fd, ...)	指定したメッセージを転送する。
S-D (a)	メッセージファイルをunlink()	受信したメッセージを破棄する。
S-E	close(socket)	メッセージポートを削除しメッセージチャネルを切断する。





	システム制御プログラムでの 処理	対応するカーネル内部の処理(メッセージチャネルインタフェース)
S201	socket = socket(...)	ソケットを作成。メッセージチャネルの接続要求を受けるための準備をする。
S202	bind(socket, ..)	システム制御プログラムが接続要求を待つためのサービスIDとソケットを対応づける。
S203	listen(socket, ...)	指定したサービスへの接続を許可する。
S204	fd = accept(socket, ...)	メッセージチャネルを接続する。
S-A (b)	read(fd, ...)	メッセージを受信して対応する(プロセスアドレス空間にマップされた)アドレスを得る。
S-B (b)	メッセージに対するアクセス	メッセージの内容を直接読み書き。
S-C (b)	write(fd, ...)	指定したメッセージを転送する。
S-D (b)	メッセージの領域を開放	プロセスのアドレス空間へのマップを解除する。
S-E	close(socket)	メッセージポートを削除しメッセージチャネルを切断する。



【要約】

【課題】 オペレーティングシステム（OS）間のスムーズなメッセージ転送を実現する装置、方法を提供する。

【解決手段】 マルチOS環境において、物理アドレス空間のメッセージ領域をメッセージ送信側OSの論理パーティションアドレス空間のメッセージ領域から、メッセージ受信側OSの論理パーティションアドレス空間のメッセージ領域へマッピング状態を切り替える処理を実行してOS間のメッセージ転送制御を行なう。また、ファイルディスクリプタに関連付けられたソケットを生成し、ソケットを介してアクセス可能な仮想ファイルを設定し、仮想ファイルと物理アドレス空間のメッセージ領域のマッピングを適用したメッセージ送受信を行なう。

【選択図】 図7

0 0 0 0 0 2 1 8 5

19900830

新規登録

5 9 7 0 6 2 9 9 3

東京都品川区北品川 6 丁目 7 番 3 5 号

ソニー株式会社

Document made available under the Patent Cooperation Treaty (PCT)

International application number: PCT/JP2005/017648

International filing date: 26 September 2005 (26.09.2005)

Document type: Certified copy of priority document

Document details: Country/Office: JP
Number: 2005-152788
Filing date: 25 May 2005 (25.05.2005)

Date of receipt at the International Bureau: 15 November 2005 (15.11.2005)

Remark: Priority document submitted or transmitted to the International Bureau in compliance with Rule 17.1(a) or (b)



World Intellectual Property Organization (WIPO) - Geneva, Switzerland
Organisation Mondiale de la Propriété Intellectuelle (OMPI) - Genève, Suisse